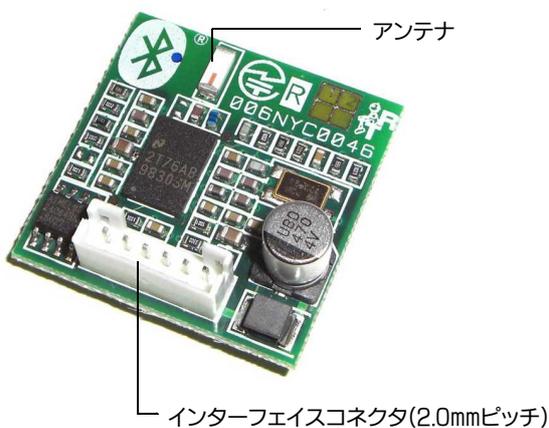
型番:RBT-001 REV.4.20

簡単! *Bluetooth(R)*-シリアルモジュール

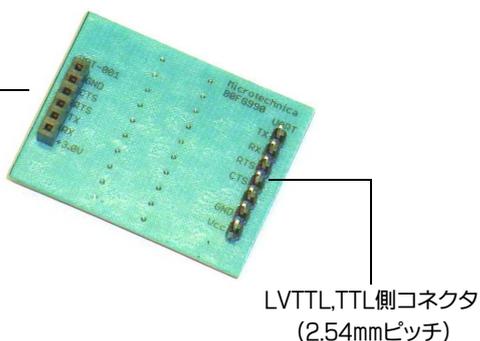
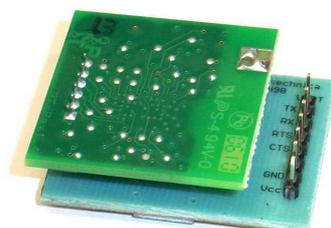
取扱説明書
お使いになる前にこの説明書をよくお読みの上正しくお使いください。本製品のサポートは製品の開発元であるイタリアRoboTech Srl社が直接行います。

製品本体

■RBT-001本体

インターフェイスコネクタ(2.0mmピッチ)

■電源電圧、電圧レベル変換アダプタ(80FG990)

RBT-001側コネクタ(2.0ミリピッチ)

LVTTL,TTL側コネクタ
(2.54mmピッチ)

■RBT-001と80FG990を組み合わせた場合

製品の概要

簡単!*Bluetooth(R)*-シリアルモジュール(型式:RBT-001、以下型式で記載)は、29×29(mm)という小さい基板内にアンテナを搭載し、近距離無線通信規格の*Bluetooth(R)*バージョン2.0に準拠した、組込用モジュールです(※1)。電波強度を規定したクラスではクラス2に対応しており、通信距離は約30mまで可能です(※2)。本体にアンテナを搭載しています。

プロファイルは、GAP・SDAP・SPPに対応しており、RS232C通信(非同期式シリアル通信=UART)を仮想化して*Bluetooth(R)*機器間で無線通信を行うことができます。UARTを無線化してパソコンなどの*Bluetooth(R)*搭載機器と簡単に通信させることができ、*Bluetooth(R)*の技術的な知識がなくても、簡単に*Bluetooth(R)*を搭載したアプリケーションを作ることができます。UART側のシリアル通信速度は最大921.6kbpsを実現。高速な通信を行うことができます。RBT-001同士の通信も可能です。

RBT-001は電波法に基づく技術基準適合証明を取得済みですので本モジュールを組み込めば、すぐに*Bluetooth(R)*を使用できます。本体には技適マークと認証番号(006NYC0046)を表示しています。お客様が面倒な申請手続きなどをする必要はありません。また、*Bluetooth(R)* SIGのライセンスも取得していますので、すぐに製品に組み込んで使用することが可能です。

RBT-001の電源電圧は+3.0Vで、ロジック信号の電圧レベルも3.0Vですが、TTLの5V系回路にすぐに接続して使用できるようアダプタ(型式80FG990)もご用意しました。コネクタも2.54mmピッチに変換されますので、すぐに5V系のロジック回路と接続したいという方はこのアダプタをお使いになると便利です。

また80FG990は、当方で販売中のPICマイコン統合開発用評価ボードシリーズ各種にそのまま装着してお使い頂けます。

※1:バージョン1.xにも対応しています。
※2:障害物や周囲の電磁環境によって通信距離は変わります。

パッケージの内容

■同梱物
・RBT-001本体
・マニュアル(本書) ※
・ユーティリティーソフトウエア ※

※マニュアル及びユーティリティーソフトウエアはインターネットからのダウンロードによるご提供となります。下記サイトからダウンロードしてご使用下さい。

http://www.microtechnica.net/manual/

対応プロファイル

*Bluetooth(R)*は、色々なデバイス間での通信を目的に作られた規格ですので、どんな機器に使えるのかということを策定してるプロファイルという標準規格があります。*Bluetooth(R)*では、通信相手の機器と同じプロファイルを持つ機器間でのみ、通信を行うことができます。
RBT−001では下記のプロファイルに対応しています。

・GAP(Generic Access Profile)
機器との接続や認証を行うためのプロファイルです

・SDAP(Service Discovery Application Profile)
*Bluetooth(R)*機器が提供する機能を調査するプロファイルです

・SPP(Serial Port Profile)
仮想シリアルポートを作り、UARTを*Bluetooth(R)*で無線化するプロファイルです。

インターフェイス端子の概要

ピン	端子記号	内　容
1	Vcc	電源ピン　+3.0V　(±5%以内)　要安定化
2	RX	UART　受信データ
3	TX	UART　送信データ
4	RTS	UART　送信要求　(アクティブロー)
5	CTS	UART　受信準備完了　(アクティブロー)
6	GND	電源ピン　GND

コネクタのピンピッチは2.0mmピッチです。

電源について

RBT−001の電源電圧は、+3.0Vです。許容誤差は±5%以内です。仕様上の絶対最大定格は+3.3Vですが、電波の電界強度の関係上3.0Vにすると最もパフォーマンスがよくなります。
電源には安定化された電源をご使用下さい。リプルの重畳している電源は使えません。スイッチング電源やスイッチングレギュレターを使用している場合、出力電圧の波形に発振波形が乗っていることがありそれらがパフォーマンスに影響を及ぼすことがあります。設計に際しては電源品質について十分ご留意下さい。
下記はスイッチングレギュレター、MC34063を使用した回路例です。入力電圧に関わらず3.0Vの電源を取り出せます。(詳しくはMC34063のデータシートをご覧下さい。)

消費電流は最大40mA程度です。(詳しくは電気的仕様の項目をご覧下さい。)十分な電流が取り出せる電源をご用意下さい。

UART信号について

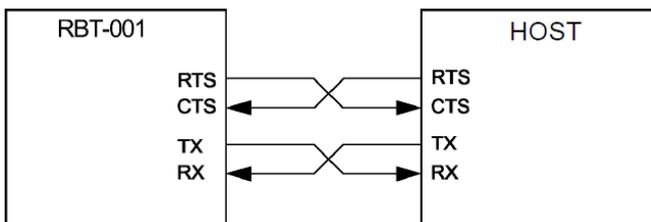
■ロジック信号の電圧レベルについて
Vccが3.0Vの時、UART信号電圧レベルは0V−3.0Vとなりますが、電気的仕様は下記の通りです。

・ViH[ロジック信号Hレベル入力電圧]　(Vcc=3.0V時)
最小値+2.1V〜最大+3.2V
・ViL[ロジック信号Lレベル入力電圧]　(Vcc=3.0V時)
最小値−0.2V〜最大0.75V

RBT−001の電圧レベルは3.0Vのため、5V系の回路や3.3V系の回路とは直結できません。(直結した場合RBT−001が破損します)当方では、簡単に5V系の回路と接続できる電源電圧、電圧レベル変換アダプタ(80FG990)を販売していますので、自作回路を作らない場合にはこちらをご使用下さい。回路を自作される場合には、GTL2002のような電圧レベル変換ICを使います。下図に参考回路を掲載します。

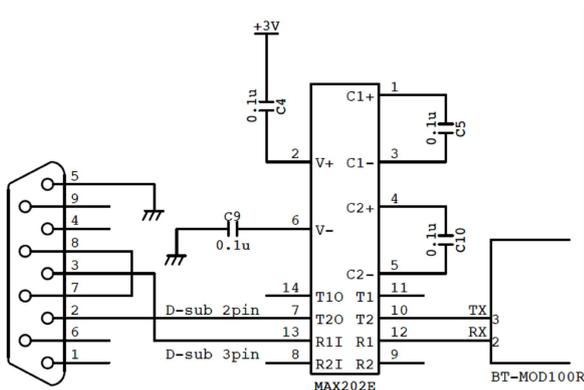
PICマイコンなどのマイコンと接続する場合には、マイコン側のロジック信号の電圧レベルが3.0Vp-p以外の場合には必ず上記のような電圧レベルを適正にするための回路を介する必要があります。

■CTS線及びRTS線について

RBT-001とUART機器(例えばマイコン等)の基本的なUART接続は4線式のハンドシェイク通信となります。

TXとRX、RTSとCTSをクロスして配線することで、UART機器との通信ができます。UART通信ではTXとRXだけの2本のデータ線だけで通信を行うこともできますが、その場合には、バッファの空き状態に応じた通信制御はできません。2線式で通信を行う場合には、バッファがフルになってしまわないよう、RBT-001にデータを溜めずに、データを逐次読み出す必要があります。また、RBT-001のCTSピンとRTSピンを短絡(ショート)させてご使用下さい。

ハンドシェイク通信を行う場合、CTS(送信可能)線及びRTS(送信要求)線で制御を行います。RBT-001内部のバッファが少なくなると、RTS線によりホスト側に送信の要求を停止します。

■RBT-001をパソコンのRS232Cポートと接続する場合

パソコンのRS232Cポートと接続する場合には、レベル変換ICを介して接続します。RBT-001のロジック電圧レベルは0V-3Vなので、レベル変換ICの電源電圧は必ず+3.0Vとします。配線例を下図に示します。下図は2線式通信の場合の例です。

RBT-001のデータ転送モード

RBT-001には2つの転送モードがあります。
1つは"コマンドモード"、もう1つは"トランスペアレントモード"です。

■コマンドモード

コマンドモードは、主にシリアルコマンドによるRBT-001の設定や簡単なデータ転送を行うモードです。データはバースト転送されずに決まったパケットフレームによって転送されます。主にコマンドを送信ためのモードです。デフォルト設定の場合には、*Bluetooth(R)*リンク確立前はコマンドモードになっており、各種シリアルコマンドによる設定可能状態となっており、リンク確立後はデータ通信用に次に述べるトランスペアレントモードに移行します。(リンク確立後もUARTブレークすることでRBT-001はコマンドモードに移行します。)

コマンドモードで使われるパケットフレームは下図の通りです。

スタートﾃﾞﾘﾐﾀ	ﾊﾟｹｯﾄﾀｲﾌﾟ定義	ｵﾍﾟｺｰﾄﾞ	ﾃﾞｰﾀ長	ﾁｪｯｸｻﾑ	ﾊﾟｹｯﾄﾃﾞｰﾀ	ｴﾝﾄﾞﾃﾞﾘﾐﾀ
0x02	1ﾊﾞｲﾄ	1ﾊﾞｲﾄ	2ﾊﾞｲﾄ	1ﾊﾞｲﾄ		0x03

チェックサム

すべてのシリアルコマンドは上記のパケットフレームに即して送信します。詳しくは本書の"シリアルコマンドの基本フォーマット"の項をご覧下さい。

■トランスペアレントモード

トランスペアレントモードは1つのリモートデバイスとリンクを確立後、高速にデータ転送を行うデータ転送専用のモードです。

通信は全二重で行われ、コマンドモードのような決まったパケットフレームはありません。通信は高速でバースト的に行われるため、バッファの空き状態を常に監視しながら通信を継続する必要があり、このモードを使用する場合には原則としてCTS/RTSを用いたハンドシェイク通信を行います。ハンドシェイク通信を行わず2線式通信でもトランスペアレントモードで通信をすることは可能ですが、バッファの制御信号がないため、バッファがいっぱいになるとデータの欠落が発生する場合があります。通信速度が速く、受信したデータを次々とRBT-001から取り込めないUART機器を使用した場合には、データの取りこぼしが発生することがあり注意が必要です。

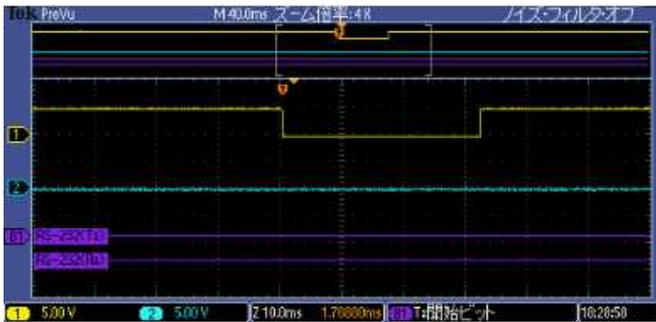
トランスペアレントモードでは、通信はすべて1対1の通信となります。基本的にはパソコン1台とRBT-001が1台という組み合わせになります。

トランスペアレントモードではRS232Cのブレーク信号を受信すると"UARTブレーク"状態となります。UARTブレークについては後述します。

トランスペアレントモードで、パソコンの仮想COMポートと通信を行う場合、仮想COMポートをオープンするまでに多少の時間がかかります。RBT-001はトランスペアレントモードが開始されると、"SPP_INCOMMING_LINK_ESTABLISHED"が発行されます。UART側の機器はこれを受信した後から通信を開始するように設計を行ってください。なお、イベントレポートを無効にしていると、"SPP_INCOMMING_LINK_ESTABLISHED"が受信できませんので注意が必要です。

■UARTブレーク

UARTブレークとは、一般的なRS232C通信でいうところのブレーク信号と同義です。RBT-001は、特に"No UART Break"の設定をしていなければ(コマンド一覧の"SET EVENT FILTERの項参照")ブレーク信号を受信すると、直ちに"UARTブレーク状態"となります。
この状態は、*Bluetooth(R)*での仮想COMポートとのリンクを保ちながら、データ通信だけを停止する状態です。コマンドモードと同様にシリアルコマンドが送受信できるようになります。後に紹介する低消費電力モードに移行させる場合などに使用します。

UARTブレーク信号は、スタートビット及びストップビットがない、電位レベルが0Vの状態で、連続して1バイト分の長さをこの状態とすることでブレーク信号と認識されます。一般的には30ミリ秒～数百ミリ秒を電位0Vとします。(途中で電圧が発生して電位が高くなるとブレーク信号ではないと判断されてしまいます。)下記に理想的なブレーク信号の波形を表示します。

上図では、約32ms程度の0を送信しています。

UARTブレーク信号をRBT-001が受信すると直ちにトランスペアレントモードはデータの送受信を停止し、コマンドモードと同等のコマンド受信期間に入ります。但しここで通常のコマンドモードと大きく違う点は、仮想COMポートはオープンされた状態のままであるという点です。

UARTブレークから再度、トランスペアレントモードに復帰する場合には、"SPP_TRANSPARENT MODE"を実行します。

■コマンドモードとトランスペアレントモードの切り替わるタイミング

コマンドモードとトランスペアレントモードは、シリアルコマンドを用いて切り替えることも可能ですが、デフォルト設定では、RBT-001が他の*Bluetooth(R)*機器とリンクを確立していない時はコマンドモード、リンク確立後はデータ通信開始とみなしてトランスペアレントモードに移行するように設定されています。

これは、シリアルコマンドによる各種設定等はリンク確立前に行い、リンク確立後は一切、シリアルコマンドによる設定などを行わずデータ通信に専念する‥という形式です。その他、上記で解説したUARTブレーク信号でも、トランスペアレントモードを一時的にコマンドモードに切り替えることができます。

とりあえず*Bluetooth(R)*での通信機能を使用してみる

RBT-001を使う上では様々な初期設定が必要です。もっとも身近なのはUARTの通信速度やパリティビットなどです。そのほか、Bluetootでは、*Bluetooth(R)*特有の設定などもあり、その使い方を詳しく理解するにはある程度の専門的知識が必要です。

そのためまずは複雑な設定については後の項目で解説することとして、ここでは最も簡単な方法で*Bluetooth(R)*通信を行って、動作を体験することにします。パソコンと、RBT-001との間で1対1の通信を行います。コマンドモードでコマンドを設定後、*Bluetooth(R)*リンクを確立してトランスペアレントモードで*Bluetooth(R)*通信を行います。

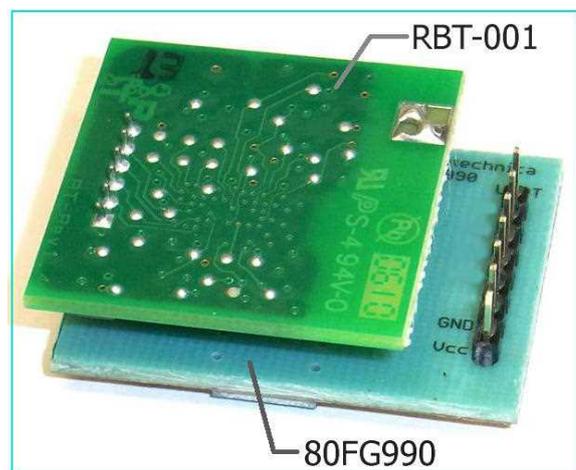
RBT-001は内部に不揮発性メモリーを搭載しており、より詳細な動作設定を本体に記憶できます。しかし使い方がやや難しいですので、とりあえず本項で解説する方法でRBT-001の使い方を体験することをお奨めします。本項で紹介する方法は、RS232C通信を*Bluetooth(R)*によって無線化するRS232Cのフルエミュレーションモードです。あたかも有線のRS232C通信を行っているかのような間隔で、*Bluetooth(R)*による無線通信が可能です。

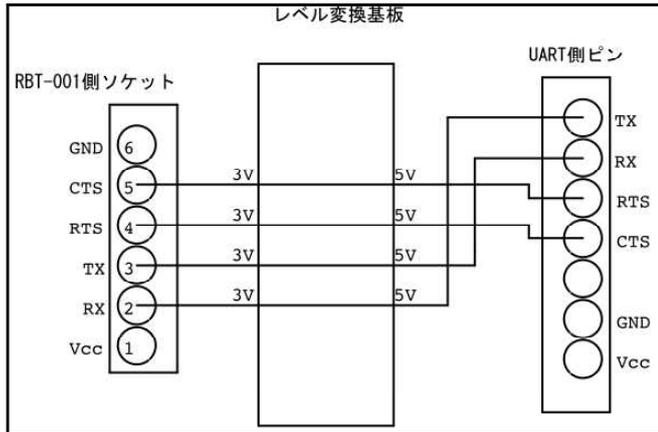
本項で解説する方法で十分であれば、このままこの方法で使用しつづけても問題ありません。

①RBT-001とUART制御機器を接続する

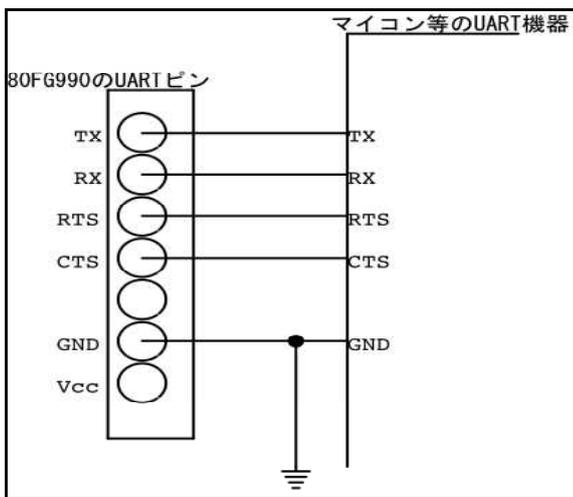
最初のRBT-001のUART側端子に、UART経由で制御しデータ通信を行いたいマイコン等の機器を接続します。インターフェイスの電気的仕様については本書2ページ～3ページに詳細が記載されていますので、必ずこちらをお読みの上、仕様を逸脱しないよう設計をお願い致します。特にロジック電圧レベルがTTLやLVTTLレベルの機器と接続する場合には、RBT-001のロジック電圧レベルは0V～+3Vですので、電圧レベルの調節が必要となります。

当方では、RBT-001への+3Vの電源電圧の給電と、ロジック電圧レベルの調整回路をワンボードに搭載したアダプタ(80FG990)を販売しております。回路の自作をされない場合には、このアダプタをご使用頂くことをお奨めします。本書では、本アダプタを使用した場合を想定した使い方を解説しています。

下図はRBT-001と80FG990アダプタを接続した場合の写真です。

※80FG990の詳細は、80FG990のマニュアルをご覧ください。

※80FG990のUART側のシルク印刷の信号線表記は、UARTホスト機器と接続するピンの表記ですので、名前通りに接続します。

RBT-001のソケットと、UARTピンの間は80FG990内で上図の通りとなっております。UART側ピンのシルク印刷のピン名表記は、UART側機器と接続する場合の接続先名称となっておりますのでご注意ください。

※UART機器のTXピンは80FG990のUARTピンのTXピンと接続、同じくUART機器のRXピンは80FG990のRXピンと接続するという意味です。CTS線、RTS線も同様です。

マイコン等と接続する場合には下図のようになります。

CTS線、RTS線は、フロー制御時に使用する信号線ですが、フロー制御をしない場合には、短絡(ショート)させておく必要があります。

②電源の給電

RBT-001のVccへ給電する電源電圧は+3.0Vです。絶対最大定格は+3.3Vとなっています+3.0Vの電源をご用意下さい。消費電流は最大で30mA程度ですので、余裕を持たせ、100mA程度は流せる電源をご用意下さい。

80FG990アダプタをご使用であれば、アダプタのVccピンに+3.3V～+5.0Vの電源を印可できます。

電源の品質は、そのままRFの性能に反映されます。よって電源品質が悪いと、通信距離が著しく短くなるなどの性能低下が発生しますので、電源にはリプルやスイッチングノイズの少ない高品質のものをお使い下さい。電源の参考回路例は本書2ページに記載しています。

③*Bluetooth(R)*機能搭載WindowsXPパソコンとの接続(ペアリング)

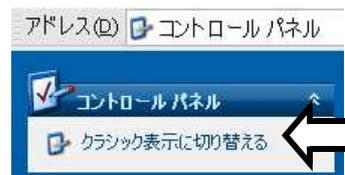
RBT-001の準備ができたので早速パソコンと無線通信をするために、接続を開始しましょう。RBT-001の周囲に*Bluetooth(R)*機能を搭載したパソコンを設置してください。*Bluetooth(R)*機能が内蔵されていないパソコンの場合には、別途*Bluetooth(R)*ドングルを取り付けて*Bluetooth(R)*機能を付加させます。(当方でもオプション品として販売しています。)

次の手順でRBT-001とパソコンとを無線接続します。なお、この無線接続を行う接続の認証のことをペアリングと言います。

以下の手順はWindowsXPを例にした解説しています。OSによって画面や設定手順が異なります。また、サードパーティー製のBluetoothスタックソフトウエアがパソコンにインストールされている場合には、そのソフトウエア上で設定する場合があります。

1 パソコンを起動します。*Bluetooth(R)*機能を有効にしておきます。

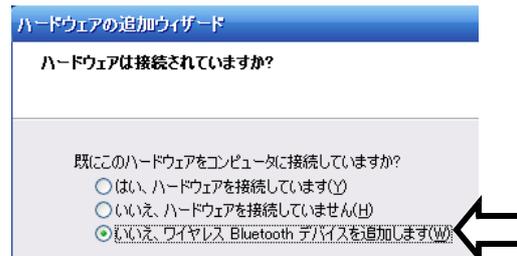
2 Windowsのコントロールパネルを開きます。*Bluetooth(R)*デバイスがインストールされている場合には、"*Bluetooth(R)*デバイス"アイコンがありますのでクリックします。

もし上図のようなアイコンが表示されていない場合には、ウインドウ左側にある"クラシック表示に切り替える"をクリックして表示を切り替えてお試し下さい。

それでも表示が出ない場合には、次の手順で*Bluetooth(R)*デバイスを手動で追加します。(WindowsXPの場合)

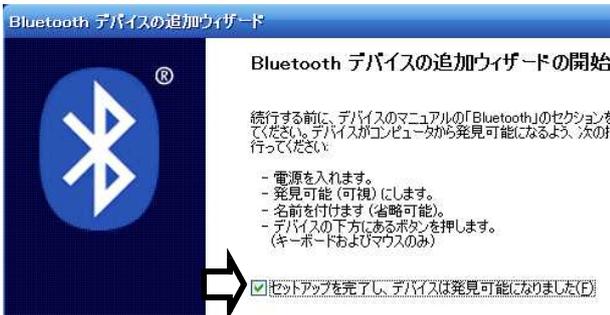
<1>コントロールパネルをクラシック表示にします。
<2>"ハードウエアの追加"をクリックします。

<3>"ハードウエアの追加ウィザード"が表示されますので、"次へ"ボタンを押して続行します。しばらく自動検索が実行されますので、完了するまで待ちます。

<4>"ハードウエアは接続されていますか?"というダイアログが表示されますので、その中から"いいえ、ワイヤレス*Bluetooth(R)*デバイスを追加します"にチェックを入れて、"次へ"ボタンを押します。

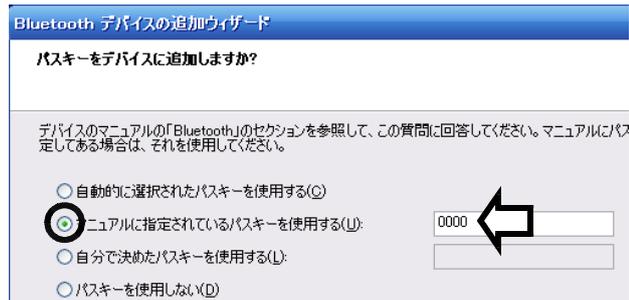
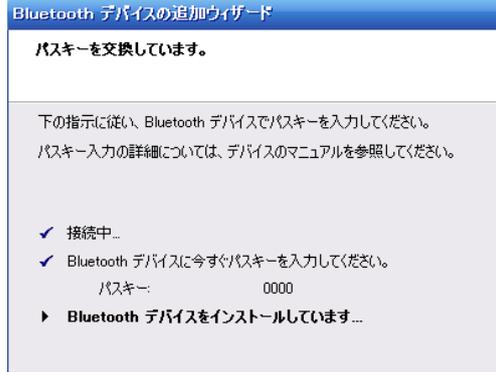
ハードウェアの追加ウィザード
ハードウェアは接続されていますか?
既にこのハードウェアをコンピュータに接続していますか?
○はい、ハードウェアを接続しています(Y)
○いいえ、ハードウェアを接続していません(H)
◉いいえ、ワイヤレス Bluetooth デバイスを追加します(W)

<5>"ハードウエアの追加ウィザードを閉じます"というダイアログが表示されますので、そのまま完了ボタンを押して閉じます。すると、*Bluetooth(R)*デバイスというダイアログが表示されます。

3 "デバイス"タブをクリックします。

4 続いて左下の"追加"ボタンをクリックします。
*Bluetooth(R)*デバイスの追加ウィザードが表示されますので、「セットアップは完了し、デバイスは発見可能になりました」にチェックを入れて、"次へ"ボタンを押します。

5 自動的に近傍にある*Bluetooth(R)*デバイスの検出を開始します。
検索が終わるまでしばらく待ちます。(1分～2分位かかります)
検索が完了すると、新しいデバイスとして"EasyBT"というデバイスが検出されます。これがRBT-001です。

6 上記の"EasyBT"アイコンをクリックしてハイライトし、"次へ"ボタンを押します。

7 続いてパスキーの入力ダイアログが表示されます。
パスキーとは*Bluetooth(R)*で接続されたお互いの機器を認識させるために初期設定する際に入力する暗証番号の様なものです。
RBT-001のデフォルト設定ではパスキーは 0000 となっています。(数字のゼロを4つ)

ウィザード内の"マニュアルに指定されているパスキーを使用する"にチェックを入れて、その右隣のテキストボックスに 0000 と入力してください。

8 RBT-001と、パソコンとのペアリンクが開始され、デバイスが認証されると、ドライバーがインストールされます。

RBT-001はSPPプロファイルに対応しているため、パソコンはRBT-001デバイスに対して、仮想COMポートを作ります。

仮想COMポートとは、実際にはパソコン上に物理的には存在しないRS232Cポートを作り、そのポートにRS232Cと同じ方法でアクセスすることで、パソコンとRBT-001が通信できるようになる便利な機能です。

ウィザードがデバイスの認識を完了すると、割り当てられたCOMポート番号がウィザードに表示されます。このポートに対してアクセスするようになりますので、必ず控えておいて下さい。

必ず控えておいて下さい。

これでパソコンとRBT-001は上記のCOMポートを介して通信を行うことができるようになりましたが、ここではまだシリアル通信のソフトウエアを起動しないでください。
シリアル通信のソフトウエアを起動して、上記のポートを開いてしまうと、*Bluetooth(R)*のリンクが確立してしまい、設定ができなくなってしまいます。

続いてRBT-001の最も基本的な設定と基本的な使い方を紹介します。

④PICマイコンとRBT−001を接続して設定を行う

細かい設定は色々とありますが、ここではまず使い方を一通り体験するためにデフォルト設定で通信をしてみましょう。*Bluetooth(R)*デバイスとのリンク確立前はコマンドモードになっています。

UART側の通信設定は、デフォルトで次のようになっています。

・通信速度:　9600bps
・データ長:　8ビット長　(LSB先頭)
・ｽﾄｯﾌﾟﾋﾞｯﾄ　1
・パリティ　なし

通信速度は921.6kbpsまで設定できますが、ここではデフォルトの値を使用します。

実際のデータ通信を始める前に、RBT−001のコマンドモード時のUARTのデータフレームについて簡単に解説します。このデータフレームは、RBT−001にコマンドを送信する場合に使用するフレームです。(データ通信のフォーマットではありません。)

スタートデリミタ	ﾊﾟｹｯﾄﾀｲﾌﾟ定義	ｵﾍﾟｺｰﾄﾞ	ﾃﾞｰﾀ長	ﾁｪｯｸｻﾑ	ﾊﾟｹｯﾄﾃﾞｰﾀ	ｴﾝﾄﾞﾃﾞﾘﾐﾀ
0x02	1ﾊﾞｲﾄ	1ﾊﾞｲﾄ	2ﾊﾞｲﾄ	1ﾊﾞｲﾄ		0x03

チェックサム

コマンドを送信する場合には、上記のバケットフレームに合わせるようにしてデータを送信します。ここでは詳しい解説は省略します。(詳細は後述します)

ここでは、*Bluetooth(R)*での無線通信をあたかもRS232Cの有線通信をしているかのように動作させる"Full cable replacement"というモードにRBT−001を設定させましょう。また、色々なイベントが発生しても通知しないよう"No events reported"の設定にします。

イベントとは、RBT−001や、その周辺機器に発生した様々な状況のことで、例えばリンクを確立したり、仮想COMポートを閉じたりという操作はイベントにあたります。RBT−001は、基本的にイベントメカニズムを採用しており、イベントに対してはほとんどの場合、応答メッセージ(コンファメーションやインジケーター)があります。これは、RBT−001の動作を詳細に把握する上で重要ですが、場合によってはそれらのメッセージが不要だったり、煩わしかったりします。

イベントの通知を無効にすることで、UART通信が遮断されることなく、RS232Cのケーブル接続と同様の感覚で*Bluetooth(R)*通信ができます。(通知がないことには不都合も伴います。)また、"Full cable replacement"モードでは、UARTブレーク信号を無視しますので、電位が0の状態が長く続いてもUARTブレーク状態には移行しません。

この設定は電源投入直後に上記のパケットフレームに即した形式で設定コマンドを送信することで設定できます。次のデータを、RBT−001に送信します。合計8バイトのデータを送信することになります。RBT−001が起動したら次のデータを上から順番に送ります。

送るデータ	解　説
0x02	ｽﾀｰﾄﾃﾞﾘﾐﾀ
0x52	ﾊﾟｹｯﾄﾀｲﾌﾟ=REQ
0x4E	ｵﾍﾟｺｰﾄﾞ=0x4E=SET EVENT FILTER
0x01	データ長は1バイト=16ビット
0x00	データ長
0xA1	BCCチェックサム 上図の範囲4バイトの和
0x03	ペイロード=3
0x03	ｴﾝﾄﾞﾃﾞﾘﾐﾀ

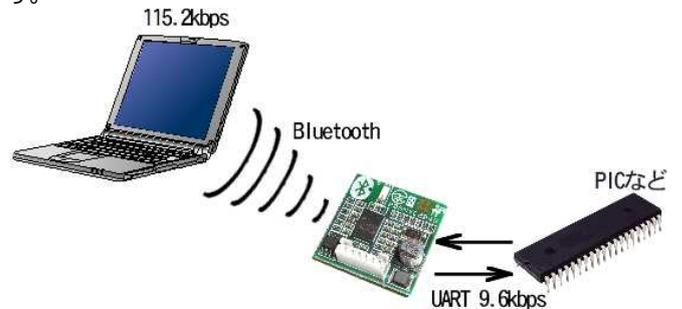
8バイトを送信後に仮想COMポートをオープンします。この仮想COMポートをオープンするタイミングが、*Bluetooth(R)*通信のリンク確立のタイミングであり、リンク確立後はすべてのUART経由のデータは、通信するデータとして見なされ、コマンドとは見なされないため、コマンドを送信する場合には、リンク確立前でなければなりません。(このコマンドだけを受け付ける期間をコマンドモード、リンク確立後のデータ通信期間をトランスペアレントモードとよんでいます。)

リンク確立後はUARTに入力されるすべてのデータは*Bluetooth(R)*で送信する送信データとして認識されます。よってコマンドを送信したつもりでも、そのコマンドも送信データとなって、送信されてしまいます。よって、コマンド送信は、仮想COMポートをオープンする前のリンク確立前に行う必要があります。

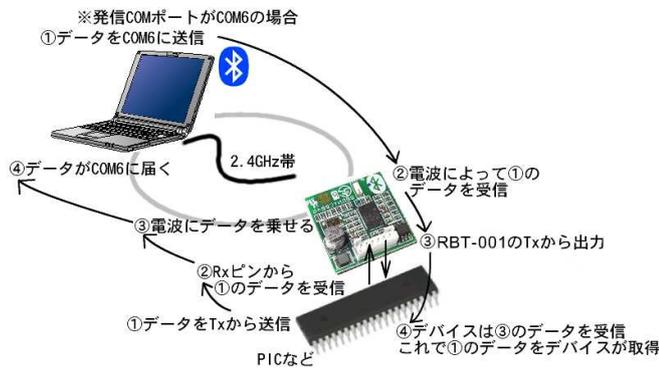
※仮想COMポートのオープンとは、パソコン側で*Bluetooth(R)*の仮想COMポートのポートを開いた状態をいいます。すなわち、ターミナルソフト等で仮想COMポートを開いた状態をいいます。

⑤パソコン側でシリアルターミナルソフトを起動する

パソコン側で、ターミナルソフト等RS232Cポートへアクセスできるソフトウエアを起動して、発信COMポートをオープンします。オープンした時点で、*Bluetooth(R)*のリンクが確立された状態となります。

仮想COMポートとターミナルソフトの間の送信速度は2400bps〜921.6kbpsまで任意に設定できます。UART側の設定には依存しません。仮にUART側の通信速度が9.6kbpsでも、ターミナルソフトと仮想COMポート間の通信速度を115.2kbpsとしても通信することができます。

ターミナルソフトで設定する、仮想COMポートの通信速度のことを、RFCOMMの通信速度と言い、UARTでの通信速度とは異なります。このRFCOMMの通信速度については基本的にRBT−001では設定する必要はありません。

⑥データ通信を開始する

パソコン側からデータを送信すると、そのデータは*Bluetooth(R)*通信でRBT-001が受信し、RXピンにデータが現れます。同様にして、RT-MOD100RのTXピンに送信したUARTデータは、*Bluetooth(R)*通信でパソコンの発信COMポートで受信できます。

※リンクの確立と切断について

パソコンとRBT-001間のリンクは、パソコン側のターミナルソフトでCOMポートを閉じるまで又は、RBT-001の電源を切断するまで続きます。

仮想COMポートを閉じるとリンクは切断されます。同様にして、RBT-001の電源を切断すると、リンクは強制的に切断されます。

パソコン側の仮想COMポートがオープンの状態で、RBT-001の電源が切断されて、リンクが切断された場合、仮想COMポートの状態は不安定になります。仮想COMポートをクローズしないで、再度RBT-001の電源を投入しても、リンクは確立できません。再度リンクを確立するためには仮想COMポートを一度クローズして、その後再度オープンすることで新しくリンクを確立することができます。

リンク確立中に仮想COMポートがクローズされた場合･･･ そのままリンクは切断される。 再度仮想COMポートをオープンするとリンクが確立される。
リンク確立中にRBT-001の電源が切断された場合･･･ リンクは切断され、仮想COMポートはデバイスを見失う。 再度BT-MOD100の電源を投入してもリンクは確立しない。 仮想COMポートを一度クローズして再度オープンすると、オープンしたタイミングでリンクが確立する。

以下に当方で販売中のPIC用CコンパイラーmikroC PRO 2009を使った場合のUART側のプログラム例を示します。デバイスはPIC16F887で、パソコン側から送信された文字(A～Zと0～9)を受信してLCDに表示します。また受信したデータはそのままパソコン側にエコーバックします。例は参考プログラムです。

```
char text;
void BT_Init();
void MPU_Init();

void main() {

     MPU_Init();
     BT_Init();

     while (1) {
        if (Uart1_Data_Ready()) {
        text = Uart1_Read();
        Uart1_Write(text);
        Lcd_Chr_Cp(text);
        }
     }
}

void MPU_Init(){
     TRISB = 0;
     ANSEL = 0;
     ANSELH = 0;

     Uart1_init(9600);
     Lcd_Init();
     Lcd_Cmd(_LCD_FIRST_ROW);
     Lcd_Cmd(_LCD_CURSOR_OFF);
     Lcd_Cmd(_LCD_CLEAR);

     return;
}

void BT_Init(){
     Uart1_Write(0x02);
     Uart1_Write(0x52);
     Uart1_Write(0x4E);
     Uart1_Write(0x01);
     Uart1_Write(0x00);
     Uart1_Write(0xA1);
     Uart1_Write(0x03);
     Uart1_Write(0x03);
     return;
}
```

上のプログラム参考例でも分かる通り、RBT-001の初期化は電源投入時又はリセット後の1回だけしか実行してません。後述しますが、RBT-001のコマンドによる設定は、*Bluetooth(R)*のリンク確立前でないと行うことができません(UARTブレーク時は除く)。

RBT-001のコマンドによる設定は、リンク確立前に行わなければならないということを覚えておいてください。

UART通信を*Bluetooth(R)*にてパソコン間とワイヤレスに簡単に使用したいという場合には、この手順で使用することが最も簡単な方法です。

"Full cable replacement"モードは、*Bluetooth(R)*通信を意識せずに有線でRS232C通信をしているかのようにユーザーが通信できるモードで最も使いやすいモードです。また、"No events reported"なので、シリアルコマンドを送信しても、それに対する戻り値がありません。

詳細な設定が必要なければ、この項に記載の方法で使用されても問題はありません。より詳しく使いたい場合には、次の項で各種設定をしてお使い下さい。

*Bluetooth(R)*によるデバイス間通信の基本

RBT-001の詳細な設定をするためには、*Bluetooth(R)*の基本的な仕組みを理解している必要があります。本項では、*Bluetooth(R)*に関する最低限の基本的な仕組みについて解説します。*Bluetooth(R)*で使われる各用語も紹介しています。なお、*Bluetooth(R)*の技術仕様等は*Bluetooth(R)* SIGがリリースしている仕様書をご覧下さい。

■*Bluetooth(R)*のペアリング

*Bluetooth(R)*によって2つのデバイスが通信を行う場合には、いずれかの1台がマスタ(又はホストと呼びます)となり、他がスレーブとなります。初期状態では全てのデバイスは待機状態になっており、自動的にその場マスターと、スレーブの関係が決まります。*Bluetooth(R)*の通信網では、1台のマスタと複数のスレーブというかたちでネットワークを構成し、この通信網のことをピコネットと呼んでいます。ピコネットには、最大1つのマスターと最大7つのスレーブがネットワークを形成します。

マスタは周囲のデバイスを探す"inquiry"という動作を行います。マスタは、他機に対してペアリングと呼ばれる相互接続の要求を行います。これを検出したスレーブデバイスは、パスキー(又はPINコード)を要求します。パスキーとは、1～16桁の英数字で、機器を固体識別するために使われる認証パスワードのことです。ペアリングするときはこのパスキー(PINコード)が必要となります。

マスタもスレーブデバイスに対してパスキーを要求し、パスキーが一致すると、ペアリングが完了し、相互に通信ができるようになります。

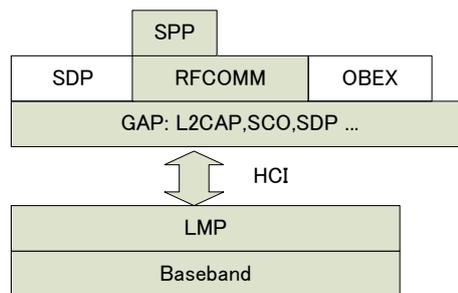
なお、マスタもスレーブデバイスも、互いの通信機器の*Bluetooth(R)* ID情報を送信し合い、自機のテーブルにそのIDを記録します。*Bluetooth(R)* IDは、機器に割り当てられた固有のIDで48ビット長です。パソコンの場合には、デバイスマネージャから*Bluetooth(R)*デバイスを選択してプロパティを閲覧すると、アドレスを知ることができます。

■*Bluetooth(R)*の接続方式

*Bluetooth(R)*の接続方式には、SCO(Synchronous Connection Oriented)と、ACL(Asynchronous ConnectionLess)という2つがあります。

SPPに対応したRBT-001は、ACL方式のみを使用します。ACLの特徴としては、1つのマスタに対して最大7つまでのスレーブデバイスと通信ができる1対多通信ができるということがあります。また、信頼性が要求されるデータ通信に適しており、パケットが壊れた場合には再送して通信品質を保つことができます。但し、通信速度は保証されておらず、上りと下りの速度は非対称となっています。

■*Bluetooth(R)*のプロトコルスタック

*Bluetooth(R)*のプロトコルスタックは様々な仕様が拡張されたため、現在は複雑になっていますが、RBT-001と関係する部分だけを主に抜粋すると次の図のようになっています。

RBT-001では、上記の色塗りの部分のプロトコルを搭載しています。GAP層の上位がRFCOMMでさらにその上位にSPPがあります。

LMP(Link Manager Protocol)層は、*Bluetooth(R)*機器間のリンクの確立やセキュリティ機能を提供しています。HCI(Host Controller Interface)は主にハードウエア部分と上位のアプリケーション部分のつなぎ役です。RBT-001ではHCIを提供していますので、開発者はハードウエアを意識せずに上位のアプリケーションの実装を行うことができます。

L2CAP(Logical Link Control and Adaptation Protocol)は、接続する機器間の伝送路の論理チャンネルを設定します。物理的な接続はBasebandが行いますが、論理的な通信路はこのL2CAPが担当します。その他、上位レイヤパケットを下位パケットに分割又はその反対に組立を行い、下位パケットへのデータ形式をユーザーが意識しなくてもいいように働きます。

■RFCOMMプロトコル

上図のプロトコルスタックのRFCOMMは、RBT-001の機能そのもののプロトコルです。

RBT-001がシリアル通信を無線化できるのは、RFCOMMがあるためです。RFCOMMはRS232C等の有線によるシリアルポート機能を*Bluetooth(R)*通信でエミュレートする機能を提供します。RFCOMMは、L2CAPレイヤ経由で下位レイヤに接続します。

SPP(Serial Port Profile)は、仮想COMポートを設定して、2台の*Bluetooth(R)*デバイス間で通信を行うプロファイルですが、このプロファイルはRFCOMMがあるからこそ実現できる機能なのです。

シリアルコマンドの基本フォーマット

先の例では、とりあえずUART通信のデータを*Bluetooth(R)*で無線通信によってパソコンの仮想COMポートの間でやりとりする例を紹介しました。この方法は、最も簡単ですぐに利用できる方法ですが、通信速度が9.6kbpsで固定されているなど・・色々と制限があります。

RBT−001をもっと自由に設定して使用したい場合には、シリアルコマンドを使って設定を変更して使用することになります。

RBT−001のUARTプロトコルのフレームは下図のように決められています。

スタートデリミタ	パケットタイプ定義	オペコード	データ長	チェックサム	パケットデータ	エンドデリミタ
0x02	1バイト	1バイト	2バイト	1バイト		0x03

←――チェックサム――→

すなわち、設定する内容が変わってもUARTコマンドで設定を行う場合には必ずこのフレームにデータが一致するようにしなければなりません。先の例ではRBT−001の初期化をするということで、下記のようにデータを送信しました。この送信順序は上記のフレームに沿った内容となっています。先の例を参考にしながら解説します。

①スタートデリミタ

通信開始の合図をする信号で、0x02に固定されています。

②パケットタイプ定義

パケットのタイプを次の4種類から定義します

コード	パケットタイプ	詳　細
0x52	REQ	RBT−001に対するリクエストのパケットです。
0x43	CFM	RBT−001がリクエストに対する確認を応答するパケットです。
0x69	IND	RBT−001から送られる情報のパケットです。
0x72	RES	問い合わせに対する応答メッセージのパケットです。

RBT−001に対してコマンドを送信する場合には、すべてリクエストですので、0x52を付けることになります。

RBT−001から送信したコマンドに対して返答がある場合には、ほとんどがCFMとなります。

コマンドの種類によってどのタイプにするのか、又はなるのかについては決まっています。

③オペコード

オペコードは、RBT−001に対してのコマンドで1バイトです。

オペコードは約80種類ありパケットデータと組み合わせて各種設定などを行います。

先の例では0x4Eを送信していますが、0x4Eは"SET EVENT FILTER"で4種類のオプションからイベント発生時（データが送受信された時のRBT−001のふるまい方）の動作方法を指定できます。

ここでは、0x03を指定しています。0x03は、*Bluetooth(R)*通信を、ケーブルでの有線通信と同じようにデータの送受信を常に継続する"Full cable replacement"というモードで、さらに"No events reported"で発生する色々なイベントに対してRBT−001の応答がないモードに設定しています。

オペコードにはよく使用するものと、使用しないものがあります。本書ではよく使用する設定についてのみ、そのオペコードを交えて解説をしています。

オペコードの一覧は本書の最後に付録として記載しています。

④データ長

バケットデータの部分で送信するデータのサイズを指定します。最大サイズは333Kバイトとなっており、2バイト長で指定します。

⑤チェックサム

Block Check Character(BCC)チェックサムを使用します。

チェックサムの計算の範囲は、パケットタイプ定義〜データ長までの4バイトです。計算方法は4バイトすべての値を加算して、その下位バイトとなります。例えば計算結果が0x3724となった場合チェックサムは0x24となります。

例えば先の例では

0x52(パケットタイプ)+0x4E(オペコード)+0x01+0x00(データ長)=0xA1となるので、0xA1をチェックサムとして送信しています。

⑥パケットデータ

送信するデータです。データ長は④で指定したデータ長でなければなりません。データの内容は様々ですが主にオペコードに対応した値となります。

⑦エンドデリミタ

通信終了の合図をする信号で、0x03に固定されています。

シリアルコマンドの基本的な決まり

オペコードを使用することで、シリアル通信経由でRBT−001の各種設定が可能です。設定はRBT−001に接続されたUART機器からコマンドを送信して行います。

シリアルコマンドのフォーマットは、前の項で説明したフォーマットに即して送信します。

■RBT−001の内部不揮発性メモリー

RBT−001内には各種設定を記憶するためのEEPROM(不揮発性メモリー)が搭載されています。不揮発性メモリーですので電源を切断しても設定内容は保持されます。シリアルコマンドでこのメモリーに対してデータの書き換えを行うことで各種動作モードの設定などを行います。不揮発性メモリーの内容は、本体起動の際に読み込まれ設定が反映されます。（動作中に変更しても再起動するまで設定の変更は反映されません）

※コマンドで内容を工場出荷時の設定に戻すことができます。

■イベントレポートについて

RBT−001は、シリアルコマンドを送信すると多くの場合、その内容に応じてレポートを返します。多くの場合は、送信したコマンドが正しく実行されたか、又はエラーとなったかという通知となります。その他、設定されたデータが正しいか確認するため、設定データの値を返すこともあります。

返される値はすべて先に紹介したパケットフォーマットに即して返されます。バケットタイプはCFM(0x43)になることが多く、コマンドに対する応答であることが確認できます。

イベントレポートは、設定によって返さなくすることもできます。先の例では、イベントレポートの設定を"No events reported"として無効に設定しました。イベントレポートを無効にするとすべての戻り値が無効となります。送ったコマンドに対する応答もなくなります。戻り値を調べる必要がなくなり煩雑ではなくなりますが、RBT−001の応答が確認で

きないという欠点もあります。システムの状況や内容によって使い分けて頂く必要があります。

例えばイベントレポートを有効にしておくと、*Bluetooth(R)*のリンクが確立すると下記のようなレポートが返ります。

02,69,50,07,00,C0,B9,6B,0B,F6,11,00,00,03
①②③④

①パケットタイプです。このパケットがRBT-001からの情報であることをを意味します。

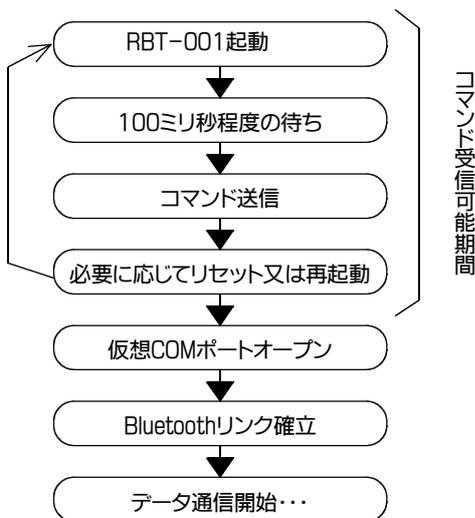
②オペコードの0x50はACLリンクが確立したことを通知します。ACLとはデータ通信用のパケットのことでSPPの場合にはこのACLパケットでデータ通信を行います。

③*Bluetooth(R)*デバイスのアドレス値です。リンク確立時に割り当てられるアドレス値で、*Bluetooth(R)*デバイスには必ず48ビットのアドレスが必ず割り当てられます。

④ステータス情報です。リンク接続にエラーがないことを通知しています。

上記の通り、イベントレポートを有効にしておくと様々な情報をRBT-001から得ることができます。しかし、例えば*Bluetooth(R)*アドレスを知る必要がないアプリケーションの場合には、この情報は意味を持ちません。イベントレポートが常にお客様のアプリケーションに意味を持つとは限りません。すべてのイベントレポートを必要としない場合には、先の例のようにイベントレポートを無効にしてしまうという方法もあります。

■シリアルコマンド送信のタイミング

シリアルコマンド送信のタイミングは、通信相手の機器とのリンクが確立する前に行います。RBT-001の通信相手は基本的にSPPプロファイルに対応した機器なので、パソコンなど仮想COMポートドライバーを搭載した機器です。リンクの確立とは、この仮想COMポートがオープンした時の状態です。パソコンの場合では、シリアルターミナルで発信ポートのCOMポートをオープンした時を指します。

■リンク確立後のコマンド

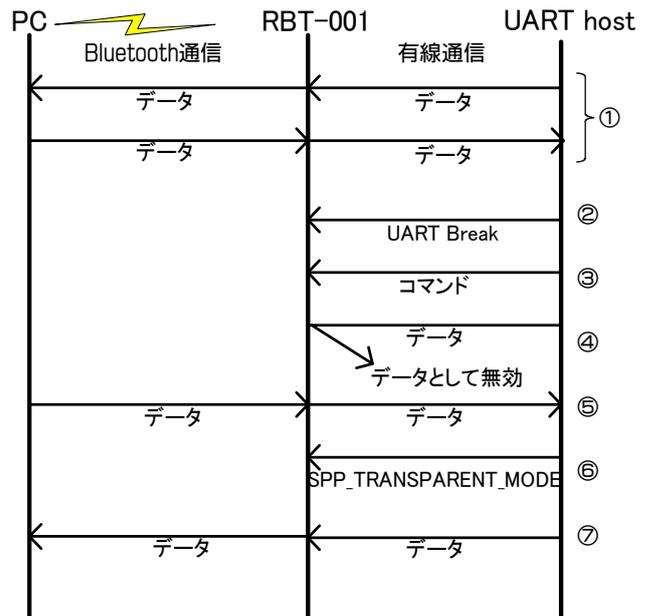
*Bluetooth(R)*のリンクが確立されると、RBT-001はトランスペアレントモードに移行します。トランスペアレントモードは、コマンドなどは一切受け付けず、すべてのデータは送受信データとして認識され、接続先の*Bluetooth(R)*機器へデータはバースト転送されます。よって、*Bluetooth(R)*リンク確立後はコマンドを送信することはできません。コマンドを送信してRBT-001の設定をするためには、仮想COMポートをクローズしてリンクを切断するか、特殊な方法としては、UARTブレーク信号を送信して、UARTブレーク状態にしなければなりません。

リンクが切断されると自動的にRBT-001は、コマンドモードに切り替わります。

■UARTブレーク中の動作

トランスペアレントモード実行中にUARTブレーク信号を受信すると、RBT-001は直ちにUARTブレーク期間になります。但し、コマンド設定で"SET EVENT FILETER(0x4E)"において、設定値0x03としてUARTブレークを無視する設定にしている場合には、その限りではありません。

UARTブレーク期間中は、RBT-001はコマンドは受け付けますが、データは*Bluetooth(R)*機器に送信しません。ただし、パソコン側の仮想COMポートは開いた状態ですので、PC側からのデータはコマンドモード中にデータを受信する"INCOMING DATA"として受信されます(SPP_INCOMING_DATAの項参照)。

ブレークモードからトランスペアレントモードに移行したい場合には、"SPP_TRANSPARENT_MODE"コマンドで移行できます。

上図は仮想COMポートの開かれたパソコンと、RBT-001が*Bluetooth(R)*で通信し、RBT-001とUART機器(マイコン等)とは有線で接続されている場合の例を示したものです。

①トランスペアレントモード実行中でデータはUARTとPCで自由に通信ができています。

②UART機器からUARTブレーク信号が発行されました。直ちにRBT-001はUARTブレークモードになります。

③UARTブレークモード中は、コマンドモードと同様にシリアルコマンドによる各種設定等が可能になります。

④データを送信しようとしても、UARTブレークモード中はデータとしては相手機器(PC側)には伝送されません。

⑤PC側から送られてくるデータは、”INCOMING DATA”として、受信されます。(データは受信できます。)

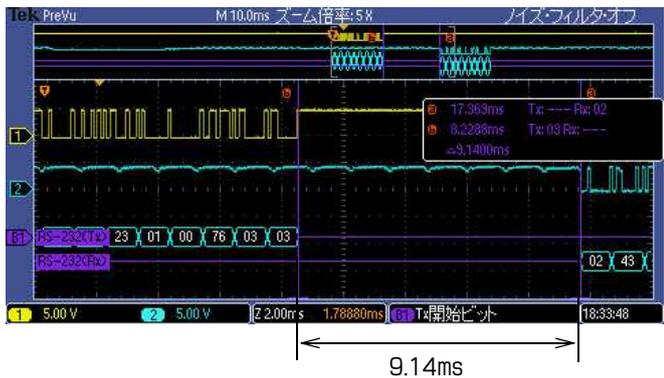
⑥UART機器から”SPP_TRANSPARENT MODE”をRBT-001が受信すると、トランスペアレントモードに移行します。

⑦トランスペアレントモードに移行後は、すべての信号は、データとして通信されます。

なお、UARTブレークモード中に”SPP_RELEASE_LINK”を実行すると、リンクは切断され⑤のPC側からのデータも受信しなくなります。リリースが完了してしまうと、パソコン側の仮想COMポートを一度クローズして再度オープンするまで、リンクは確立できません。
(”SPP_TRANSPARENT MODE”を発行してもリンク確立はできません。)

■コマンド送信後のコンファメーション(戻り値)が返る時間について

”SET EVENT FILETER(0x4E)”で、イベントレポートを有効に設定しておくと、RBT-001は受信したコマンドに応じたコンファメーションを返します。コンファメーションとは、戻り値のことで、RBT-001がリクエストに対して、どのような処理を行ったかを通知するものです。

コンファメーションが戻るタイミングはコマンドによって異なる場合がありますが、一般的には10ミリ秒程度です。下図は、UARTの通信速度を変更する”CHANGE_NVS_UART_SPEED(0x23)”を送信して、それに対するコンファメーションを受信した場合の波形です

トランスペアレントモード状態とコマンドモード状態

トランスペアレントモードは、自機と接続先機器との間でデータを自由に交換通信できるモードで、コマンドモードは相手機器とのデータ送受信ではなく、自機の様々な設定を行うための各種コマンドを送受信するためのモードであることは先にご説明した通りです。

コマンドモードからトランスペアレントモードに移行するタイミングは、先の解説の通り、

①RBT-001とパソコンが接続される場合には‥
ペアリンク確立後のパソコン側から仮想COMポートを開いた後から

②RBT-001同士の通信などの場合には‥
SPP_ESTABLISH_LINK(0x0A)をローカルデバイスに送信しリンク確立後、SPP_TRANSPARENT_MODE(0x11)を送信した時点から

開始されます。

トランスペアレントモードでデータ通信を開始する場合には、必ずトランスペアレントモードの通信路が確立されてから、データを本機のUART側に送信するようにします。トランスペアレントモードの通信路が確立された状態というのは、パソコン側から仮想COMポートを開いて通信を始めた場合には、自機(ローカルデバイス)でSPP_INCOMMING_LINK_ESTABLISHEDが発行された(本機を接続しているUART機器にSPP_INCOMMING_LINK_ESTABLISHEDが送られた)後の状態です。このコマンドをUART機器は受信した後から、データ通信を開始するようにします。

コマンドモードの際は、本機はコマンドを設定するためのモードになっており動作を停止しているわけではありません。よって、コマンドでないデータや、本機にとって意味のないデータをUART側から送信することは本機の誤作動の原因となる場合がありお奨めできません。トランスペアレントモードで通信を実行する場合には、本機と接続したUART機器は、本機がトランスペアレントモードになったことを確認してからUART通信線にデータを送信するよう設計する必要があります。

RBT-001とRFCOMM

■RFCOMMプロトコルの役割

RBT-001が、*Bluetooth(R)*通信を介して仮想COMポートで通信ができるのは、*Bluetooth(R)*のプロトコルスタックにRFCOMMがあるためです。RFCOMMプロトコルは*Bluetooth(R)*通信を行っている空間伝送部分をあたかもRS232Cの通信をしているかのようにシリアルポートをエミュレートしています。

RFCOMMプロトコルは、2台の*Bluetooth(R)*機器間で最大60個の同時接続が可能です。RBT-001では最大30の通信路を接続できます。但し、実際には1つのデバイスとは1つの通信路を結びます。

■RBT-001でのRFCOMMの役割

RBT-001が仮想COMポートによる*Bluetooth(R)*でのシリアル通信のエミュレーションができるのは、RBT-001がRFCOMMをサポートしているからに他なりませんが、トランスペアレントモードでリンクを確立するだけであればRFCOMMの存在をユーザーが意識する必要はありません。それは、仮想COMポートがパソコン側で開かれると、RBT-001は、コマンドモードから自動的にトランスペアレントモードに移行し、通信ができるためです。

しかしながら、リンクを明示的に切断したり、UARTブレークモードの時に通信を行う場合などは、RFCOMMの番号を指定しなければならないことがあります。リンクが確立しトランスペアレントモードに移行すると、RBT-001はUART機器に対してリンクが確立したことを通知する”SPP_INCOMING_LINK_ESTABLISHD”のメッセージを送信してきます。

02	69	0C	07	00	7C	接続先のデバイスの*Bluetooth(R)*ID(6バイト長)	ローカルRFCOMM番号(1バイト長)	03

※値はすべて16進数表記

接続先デバイスの*Bluetooth(R)* IDが6バイト長で返ります。この値は、明示的に接続デバイスを指定するコマンドで使用することがあるため、プログラムを作成する場合には、配列等に保存することをお奨めします。ローカルRFCOMMはRBT-001が使用するRFCOMMのポート番号です。トランスペアレントモードでは意識する必要がない場合が多いですがUARTブレークモード中に操作をする場合等ではRFCOMMの値を指定する必要がある場合がありますので、変数等に記憶するようにしておくのが一般的です。

シリアルコマンド一覧

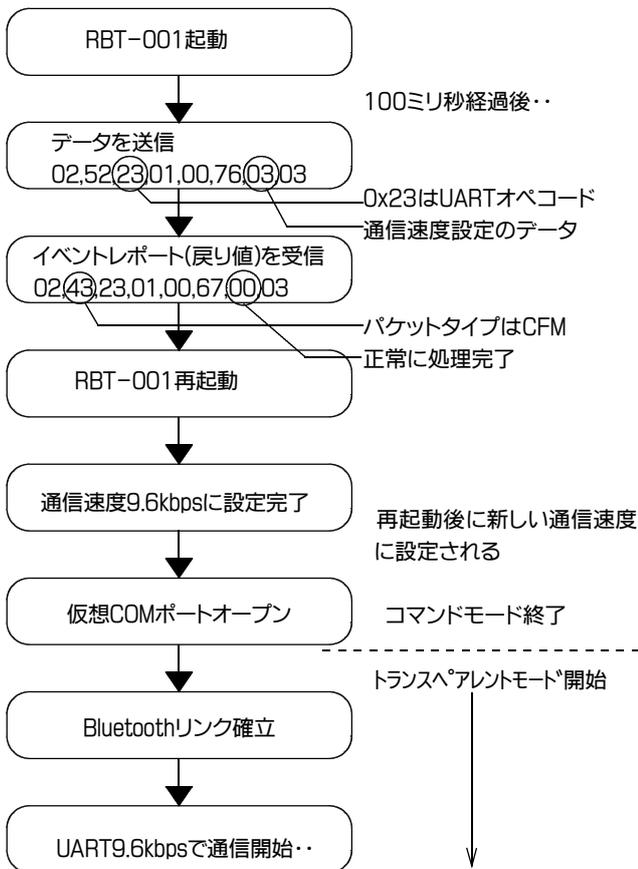
シリアルコマンドは分かりやすいよう、機能別に分けて掲載しています。"SET_EVENT_FILTER"でイベントレポートの通知を有効にしてある場合には、コマンド送信後にそのコマンドの処理結果が戻ります。この戻り値のことを、コンファメーションと呼んでいます。

コンファメーションは、基本的にパケットタイプがCFM(0x43)で、オペコードは、送信したコマンドと同じ値が返ります。

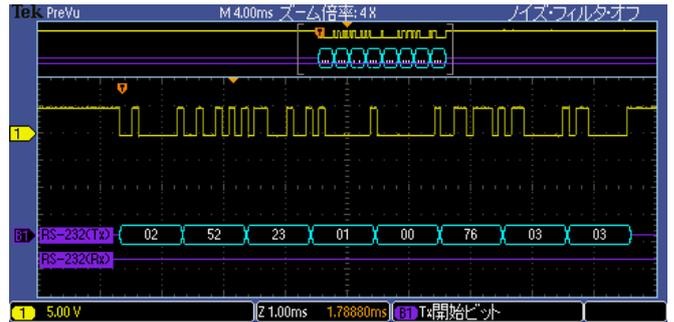
本項では特に戻り値が複雑でない場合には、コマンドと同じ枠内に"戻り値"という項目を設けて、そこに記載しています。

■シリアルコマンド送受信の例

UARTの通信速度を変更する手順を例に、シリアルコマンドの送受信例を以下に示します。(図中の値はすべて16進数です。)

例:UARTの通信速度を9.6kbpsに設定する

一連のコマンド送受信の流れは上記のようになります。
なお送信した値は、パケットフレームに合わせられており(9ページ参照)、意味は次のようになっています。

02　スタートデリミタ
52　パケットタイプはREQ(リクエストタイプ)
23　オペコード、UARTの通信速度設定の値は0x23
01　データサイズの指定、サイズはオペコードにより決められている
00　　　〃
76　チェックサム　※0x52+0x23+0x01=0x76
03　設定値のデータ、0x03は9.6kbpsにするための設定値
03　エンドデリミタ

RBT-001に送信したデータの波形を観察すると下図のようになります。

■送信コマンドと通知コマンド

コマンドのパケットタイプには、先に解説したように4つのタイプがあります。主に使用するタイプは次の3タイプです。

RBT-001に対してリクエスト(指示)を行うコマンドの場合には、パケットタイプとして0x52(REQ)を指定します。

リクエストに対する応答や、RBT-001側からコンファメーションがある場合には、パケットタイプは0x43(CFM)となります。

その他、RBT-001が現在の状態や状態が変更されたことなどを通知する場合には、0x69(IND)となります。

ﾊﾟｹｯﾄﾀｲﾌﾟ	値	詳　細
REQ	0x52	RBT-001へのリクエスト
CFM	0x43	RBT-001からの応答又は戻り値
IND	0x69	RBT-001からの状態通知

■チェックサム値の算出

チェックサムの値の算出方法は本書11ページに記載があります。チェックサムの計算の範囲は、パケットタイプ定義～データ長までの4バイトの和となります。演算結果が0xFFを超える場合には、下位の1バイトだけをチェックサムとして使用します。(例えば計算結果が0x3724となった場合チェックサムは0x24となります。)

【1】シリアル通信関連の設定

UART通信速度の設定 ※ﾃﾞﾌｫﾙﾄ設定は　9600bps	
詳細	UART通信の通信速度を10段階で設定します。 設定は不揮発性メモリーに記憶されます。 設定した内容は本体再起動後から有効になります。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	CHANGE_NVS_UART_SPEED (0x23)
ﾃﾞｰﾀ長	1
ﾃﾞｰﾀ	0x00=2400　　0x06=57600 0x01=4800　　0x07=115200 0x02=7200　　0x08=230400 0x03=9600　　0x09=460800 0x04=19200　　0x0A=921600 0x05=38400 ※単位:bps
補足	この設定速度はUART側の通信速度であり、仮想COMポートとシリアルターミナル間での速度(この速度のことをRFCOMMの通信速度と言います)ではありません。
戻り値	※戻り値とはイベントに対する結果の通知です パケット・・・　CFM(0x43) データ長・・・　1バイト ERROR_OK(エラーなし)　0x00 設定値が範囲外です　0x11 パラメーターの誤りです　0x01

UARTのパリティ設定 ※ﾃﾞﾌｫﾙﾄ設定は　パリティなし、1ストップビット	
詳細	UART通信のパリティを設定します。 設定は不揮発性メモリーに記憶されます。 設定した内容は本体再起動後から有効になります。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	CHANGE_UART_SETTINGS (0x48)
ﾃﾞｰﾀ長	2
ﾃﾞｰﾀ	1バイト目:　パリティビット 0x00=なし 0x01=偶数 0x02=奇数
	2バイト目:　ストップビット 0x00=1ストップビット 0x01=2ストップビット
補足	この設定はUART側のパリティ並びにｽﾄｯﾌﾟﾋﾞｯﾄの設定であり、仮想COMポートとシリアルターミナル間での設定ではありません。
戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト ERROR_OK(エラーなし)　0x00 パリティ設定値が範囲外です　0x2F ｽﾄｯﾌﾟﾋﾞｯﾄの設定値が範囲外です　0x30 パラメーターの誤りです　0x01

【2】*Bluetooth(R)*に関する設定

*Bluetooth(R)*のセキュリティに関する設定は、GAPプロファイルを用います。*Bluetooth(R)*に関する基本事項の設定を行います。

現在のパスキー(PINコード)の確認	
詳細	現在設定されているパスキーの値を取得します。 本コマンド送信後、約200ミリ秒以内に値が返ります。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_GET_FIXED_PIN (0x16)
ﾃﾞｰﾀ長	0

現在のパスキー(PINコード)の通知	
詳細	現在設定されているパスキーの値を返します ※パスキーのデフォルト値は0000です。
ﾊﾟｹｯﾄ	CFM (0x43)
ｵﾍﾟｺｰﾄﾞ	GAP_GET_FIXED_PIN (0x16)
ﾃﾞｰﾀ長	2+パスキーの長さ分
ﾃﾞｰﾀ	1バイト目:　ステータス情報(長さ1バイト) ERROR_OK(エラーなし)　0x00 ERROR発生　0x01
	2バイト目:　パスキーのバイトサイズ(長さ1バイト) 範囲は0x01～0x10まで
	3バイト目以降 パスキーの値が返ります(最大16バイト)
例	パスキーが0000の場合　(表記はすべて16進数) 02,52,16,00,00,68,03　UART機器で送信 ↓ 02,43,16,06,00,5F,00,04,30,30,30,30,03　戻り値 CFM　ﾁｪｯｸｻﾑ　パスキー

パスキー(PINコード)の設定	
詳細	パスキーの値を新規に設定します。 設定された値は不揮発性メモリーに記憶されます。 設定後は再起動が必要です。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_SET_FIXED_PIN(0x17)
ﾃﾞｰﾀ長	1+パスキーの長さ
ﾊﾟﾘﾃｨ値	不定
ﾃﾞｰﾀ	1バイト目:　新たに設定するパスキーの長さをバイト単位で指定します。(1バイト長) 範囲:0x01～0x10
	2バイト目以降: 設定するパスキーの値を指定します。 最大128ビット(16バイト)までです。
例	パスキーを1234とする場合　(表記はすべて16進数) 02,52,17,06,00,6F,05,30,31,32,33,34,03 REQ　ﾁｪｯｸｻﾑ　新しいパスキー

↓次のページに続きます

戻り値	パケット･･･　CFM(0x43) データ長･･･　1バイト ERROR_OK（エラーなし）　0x00 パラメーターの誤りです　0x01 パスキーの長さが不正です　0x2E

ペアリング確立済みデバイスの*Bluetooth(R)* ID取得 ※*Bluetooth(R)* IDを不揮発性メモリー内のテーブルから読み取り	
詳細	ペアリング確立済みデバイスの*Bluetooth(R)* IDを、BT-MOD100Rの不揮発性メモリーから取得します。 RBT-001は、初回ペアリング時に相手機器の*Bluetooth(R)* IDを不揮発性メモリーに記憶し、次回以降同じデバイスとは認証手順を省略します。
ﾊﾟｹｯﾄ	REQ（0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_LIST_PAIRED_DEVICE(0x1C)
ﾃﾞｰﾀ長	0

ペアリング確立済みデバイスの*Bluetooth(R)* IDの通知	
詳細	上記のコマンドで要求された、ペアリング確立済みデバイスの*Bluetooth(R)* IDを通知します。
ﾊﾟｹｯﾄ	CFM（0x43)
ｵﾍﾟｺｰﾄﾞ	GAP_LIST_PAIRED_DEVICE(0x1C)
ﾃﾞｰﾀ長	2+6×デバイス数
データ	1バイト目:　ステータス情報（長さ1バイト） ERROR_OK（エラーなし）　0x00 ERROR発生　0x01
	2バイト目:　ペアリング確立済みデバイスの個数（長さ1バイト）　範囲は0x00～0x07 ※0の場合にはペアリング確立済みのデバイスがないことを示します。
	3バイト目以降: *Bluetooth(R)* IDが6バイト長で返ります。
例	ペアリング確立済みのデバイスとして、*Bluetooth(R)* IDが 01:02:03:04:05:06の機器の場合 02,43,1C,08,00,67,00,01,06,05,04,03,02,01,03 *Bluetooth(R)* ID

登録済み*Bluetooth(R)* IDの削除	
詳細	不揮発性メモリーに記憶されているペアリング確立済みデバイスの*Bluetooth(R)* IDを削除します。
ﾊﾟｹｯﾄ	REQ（0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_REMOVE_PAIRING(0x1B)
ﾃﾞｰﾀ長	6
データ	削除したい*Bluetooth(R)* IDを6バイトで指定します。 ※下位バイトが先頭になります
補足	*Bluetooth(R)* IDの登録を削除してしまうと、次回接続時は再度ペアリング（接続の認証）の作業が必要です。
戻り値	パケット･･･　CFM(0x43) データ長･･･　1バイト ERROR_OK（エラーなし）　0x00 パラメーターの誤りです　0x01 存在しないIDを指定しています　0x0A

自機の*Bluetooth(R)* ID取得	
詳細	自機に設定されている*Bluetooth(R)* IDを取得します。
ﾊﾟｹｯﾄ	REQ（0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_READ_LOCAL_BDA(0x05)
ﾃﾞｰﾀ長	0

自機の*Bluetooth(R)* ID通知	
詳細	自機に設定さている*Bluetooth(R)* IDを通知します。
ﾊﾟｹｯﾄ	CFM（0x43)
ｵﾍﾟｺｰﾄﾞ	GAP_READ_LOCAL_BDA(0x05)
ﾃﾞｰﾀ長	7
データ	1バイト目:　ステータス情報（長さ1バイト） ERROR_OK（エラーなし）　0x00 パラメーターの誤りです　0x01 予期しないエラーです　0x05
	2バイト目以降:　*Bluetooth(R)*（長さ6バイト） *Bluetooth(R)* IDが6バイト長で返ります。

【3】*Bluetooth(R)*通信及びSPPに関する設定、コマンド

SPP incoming(発信) リンク確立の通知	
詳細	SPPによって、仮想COMポートがオープンされ、パソコンとRBT-001のリンクが確立したことを通知します。 ※この通知はイベントレポートを有効に設定しておくと、リンク確立直後に必ず返ります。
ﾊﾟｹｯﾄ	IND (0x69)
ｵﾍﾟｺｰﾄﾞ	SPP_INCOMING_LINK_ESTABLISHED(0x0C)
ﾃﾞｰﾀ長	7
データ	1バイト目～6バイト目: 接続先機器の*Bluetooth(R)* ID (長さ6バイト) リンクを確立した相手のデバイスの*Bluetooth(R)* IDが通知されます。
	7バイト目: ローカルRFCOMMポート番号(長さ1バイト) RBT-001が使用するRFCOMMプロトコルレベルでのポート番号です。範囲は0x01～0x30です。

SPP リンク切断の通知	
詳細	リンクが切断された時に通知されます。 データリンク制御(DLC)が開放されたことを意味します。
ﾊﾟｹｯﾄ	IND (0x69)
ｵﾍﾟｺｰﾄﾞ	SPP_LINK_RELEASED(0x0E)
ﾃﾞｰﾀ長	2
データ	1バイト目: リンク切断の要因 (長さ1バイト) リンクが切断された要因を下記のパラメーターで通知します。 0x00 ローカルデバイスが切断しました 0x01 接続先デバイスが切断しました 0x02 ACLでエラーが発生して切断されました 0x03 L2CAP層以下で切断要求があり切断されました
	2バイト目: ローカルRFCOMMポート番号(長さ1バイト) RBT-001が使用するRFCOMMプロトコルレベルでのポート番号です。範囲は0x01～0x30です。
補足	リンクが切断された際には、"SPP_LINK_RELEASED"のインジケーターに先行して、トランスペアレントモードが切断された旨を示す"SPP_TRANSPARENT_MODE"のインジケーターも返ります。詳しくは次の"SPP_TRANSPARENT_MODE"をご覧下さい。

SPP トランスペアレントモード終了の通知	
詳細	RBT-001のトランスペアレントモードが終了されたことを通知します。 トランスペアレントモードが終了すると、コマンドモードに移行します。
ﾊﾟｹｯﾄ	IND (0x69)
ｵﾍﾟｺｰﾄﾞ	SPP_TRANSPARENT_MODE(0x11)
ﾃﾞｰﾀ長	2
データ	1バイト目: ローカルRFCOMM番号(長さ1バイト) 範囲は1～30
	2バイト目: 移行するモード 0x00 コマンドモード 0x01 トランスペアレントモード

SPP トランスペアレントモード開始の設定	
詳細	UARTブレーク信号により、一時的に中断したトランスペアレントモードを開始する時に使用します。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	SPP_TRANSPARENT_MODE(0x11)
ﾃﾞｰﾀ長	1
データ	1バイト目: ローカルRFCOMM番号(長さ1バイト) 範囲は1～30を指定します。
戻り値	パケット・・・ CFM(0x43) データ長・・・ 2バイト 1バイト目: エラー通知 ERROR_OK(エラーなし) 0x00 SPPポートが開けません 0x21 SPPポートの値が不正です 0x20 複数のデバイスと通信を試みました 0x23 接続できません 0x1F
	2バイト目: ローカルRFCOMM番号(長さ1バイト) 範囲は1～30です。開かれたポート番号が返ります。

SPP INCOMING LINKの切断	
詳細	UARTブレークモード中に、このコマンドを発行すると、SPPのリンクは強制的に切断され、パソコンとの通信はできなくなります。また、"SPP_TRANSPARENT_MODE"を発行しても、トランスペアレントモードには復帰できなくなります。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	SPP_RELEASE_LINK(0x0D)
ﾃﾞｰﾀ長	1
データ	1バイト目:　切断するRFCOMM番号(長さ1バイト) 範囲は1～30を指定します。
戻り値	パケット・・・　CFM(0x43) データ長・・・　2バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 SPPポートが開けません　0x21 SPPポートの値が不正です　0x20 接続できません　0x1F
	2バイト目:　切断したRFCOMM番号(長さ1バイト) 範囲は1～30です。切断されたRFCOMM番号が返ります。
補足	本コマンドでSPPのリンクを強制的に切断すると、PC側からのインカミングデータ(RBT-001に送られてくるデータ)を受信しなくなります。 "SPP_TRANSPARENT_MODE"を発行しても、トランスペアレントモードには復帰できなくなります。

SPP UARTブレークモード中のデータ転送	
詳細	UARTブレークモード中に、指定したRFCOMMと通信を行っている*Bluetooth(R)*機器に対し、データを送信するコマンドです。UARTブレーク中のデータ転送では、シリアルコマンドの中に送信したいデータを入れて送信します。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	SPP_SEND_DATA(0x0F)
ﾃﾞｰﾀ長	3+ペイロードサイズ
データ	1バイト目:　ローカルRFCOMM番号(長さ1バイト) 範囲は1～30を指定します。最初に"SPP_INCOMING_LINK_ESTABLISHD"が確立した時に通知されたRFCOMM番号を指定します。
	2バイト目:　ペイロードサイズ(長さ2バイト) 送信するデータのサイズを1～330バイトの範囲で指定します。下位バイトが先になります。 例えば10バイトの場合には、 0A,00 と指定します。
	4バイト目以降;　送信するデータ　(最大330バイト) 送信するデータを指定します。データサイズは2バイト目で指定したサイズにしなければなりません。

↓次に続きます

戻り値	パケット・・・　CFM(0x43) データ長・・・　2バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 データサイズが制限に達しています　0x1B データを送信できませんでした　0x1D 接続ができません　0x1F ポート指定が無効です　0x20 指定したポートを開けません　0x21
	2バイト目:　使用したRFCOMM番号 範囲は1～30です。
補足	データの転送は本来トランスペアレントモード実行中に行うことが望ましいですが、トランスペアレントモードではデータをバースト転送するため、UART機器とハンドシェイク通信を行っていない場合、データのサイズや速度によってはデータの欠落が発生する場合があります。 このSPP_SEND_DATAコマンドを使用すると、パケットフレームにデータを格納できるため、データの転送をより確実に行うことができます。

SPP UARTブレークモード中のデータ受信	
詳細	UARTブレークモード中に、RFCOMMの接続が確立している*Bluetooth(R)*機器からのデータを送信すると、そのデータ内容を出力します。
ﾊﾟｹｯﾄ	IND(0x69)
ｵﾍﾟｺｰﾄﾞ	SPP_INCOMING_DATA(0x10)
ﾃﾞｰﾀ長	3+ペイロードサイズ
データ	1バイト目:　ローカルRFCOMM番号(長さ1バイト) 範囲は1～30です。通信に使用されたRFCOMM番号が返ります。
	2バイト目:　ペイロードサイズ(長さ2バイト) 受信したデータのサイズを1～330バイトの範囲で通知します。下位バイトが先になります。
	4バイト目以降:　受信したデータ(最大330バイト) 受信したデータを出力します。最大サイズは330バイトまでとなります。

GAP 他RBT-001デバイスの検出

詳細	通信可能な範囲内に存在する別のRBT-001デバイスの検出を行います。指定した時間内に検出すると、そのデバイスの*Bluetooth(R)* IDとクラス番号を返します。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_INQUIRY(0x00)
ﾃﾞｰﾀ長	3
ﾃﾞｰﾀ	1バイト目:　検索時間の指定(長さ1バイト) 検索時の時間を1.28秒～61.44秒の間で指定します。指定できる値は0x01～0x30です。(1を指定すると1.28秒となります。) 実用的な設定目安時間は5～10程度です。
	2バイト目:　結果通知の最大値(長さ1バイト) 検索結果の最大通知数を0x00～0xFFで指定します。0に設定すると制限なく検出したデバイスすべての結果を返します。通常は0に設定します。
	3バイト目:　0に設定します(長さ1バイト)
戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 指定間隔が範囲を逸脱しています　0x02 3バイト目の値が正しくありません　0x03 指定したパラメータが無効です　0x01

GAP 他RBT-001デバイスの検出結果の通知

詳細	GAP_INQUIRYコマンドを受信後、デバイスを検出した場合、その結果を、検出したデバイスの*Bluetooth(R)* IDと共に通知します。
ﾊﾟｹｯﾄ	IND(0x69)
ｵﾍﾟｺｰﾄﾞ	GAP_DEVICE_FOUND(0x01)
ﾃﾞｰﾀ長	9
ﾃﾞｰﾀ	1バイト目～6バイト目:　接続先機器の*Bluetooth(R)* ID(長さ6バイト) 発見した相手のデバイスの*Bluetooth(R)* IDが通知されます。
	7バイト目～9バイト目:　デバイスのクラスID(長さ3バイト) 発見した相手のデバイスのクラスIDを返します。

SPP リンクを確立させる

詳細	SPPリンクを通信相手方機器の*Bluetooth(R)* IDを指定して、手動で通信を確立します。 RBT-001同士の通信(対パソコンとの通信ではない場合)の場合に接続を確立するために使用します。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	SPP_ESTABLISH_LINK(0x0A)
ﾃﾞｰﾀ長	8
ﾃﾞｰﾀ	1バイト目:　ローカルRFCOMM番号(長さ1バイト) 範囲は1～30を指定します。 どのデバイスとも通信していない場合には、通常1から指定していきます。
	2バイト目～7バイト目: *Bluetooth(R)* ID(長さ6バイト) 通信を確立する相手方デバイスの*Bluetooth(R)* IDを指定します。*Bluetooth(R)* IDはあらかじめ相手方デバイスのIDを調べておくか、又はGAP_INQUIRYコマンドで知ることもできます。 *Bluetooth(R)* IDはリトルエンディアン(最下位バイトから順に)で指定します。 例:*Bluetooth(R)* IDが 00:17:A0:01:02:03 の場合 0302010A1700　と指定します。
	8バイト目:　リモートRFCOMM番号(長さ1バイト) 範囲は1～30を指定します。 通信する相手デバイスの空きRFCOMM番号を指定します。通常はSDAPでその値を取得することが原則ですが、簡易的には適当な値を指定します。 ※リモートデバイス(通信相手方のRBT-001)が別のデバイスとRFCOMMを使用して通信している場合、この値が正しくないと通信ができない場合があります。
補足	本コマンドでSPPのリンクが指定したデバイスとの間で官立されると、CFMで通知が返る他に、SPP_PORT_STATUS_CHANGED(0x3E)及び、SPP_LINK_ESTABLISHED(0x0B)がパケットタイプIND(0x69)で返ります。 また、リンクが確立されたリモートデバイス先では、SPP_INCOMING_LINK_ESTABLISHED(0x0C)が返ります。
戻り値	パケット・・・　CFM(0x43) データ長・・・　2バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 SPPポートが開けません　0x21 SPPポートの値が不正です　0x20 SPPポートがビジーです　0x22 すでに接続が確立されています　0x26
	2バイト目:　ローカルRFCOMM番号(長さ1バイト) 範囲は1～30です。開かれたポート番号が返ります。

SPP リンクの確立通知	
詳細	SPP_ESTABLISH_LINK発行後に、SPPリンクが確立した場合に返ります。
ﾊﾟｹｯﾄ	IND(0x69)
ｵﾍﾟｺｰﾄﾞ	SPP_LINK_ESTABLISHED(0x0B)
ﾃﾞｰﾀ長	9
データ	1バイト目: RFCOMMに関するエラー通知(長さ1バイト) エラーなし、正常処理 0x00 DLCが存在しません 0x01 ポートの設定が正しくありません 0x02 DLC接続の確立に失敗しました 0x03 アクセスを拒否されました 0x04 リモートデバイス接続に失敗しました 0x05
	2バイト目~7バイト目: リモートデバイスの*Bluetooth(R)* ID (長さ6バイト) 接続先リモートデバイスの*Bluetooth(R)* IDです。
	8バイト目: ローカルデバイスRFCOMM番号 (長さ1バイト) ローカルデバイスのRFCOMM番号を1~30の範囲で通知します。
	9バイト目: リモートデバイスRFCOMM番号 (長さ1バイト) リモートデバイスのRFCOMM番号を1~30の範囲で通知します。

SPP RFCOMMの状態が変化したことの通知	
詳細	ローカルのRBT-001のRFCOMMのポート状態が変化した場合に、その内容を通知します。
ﾊﾟｹｯﾄ	IND(0x69)
ｵﾍﾟｺｰﾄﾞ	SPP_PORT_STATUS(0x3E)
ﾃﾞｰﾀ長	4
データ	1バイト目: ローカルデバイスRFCOMM番号 (長さ1バイト) ローカルデバイスのRFCOMM番号を1~30の範囲で通知します。
	2バイト目: ポートの状態 (長さ1バイト) ポートの状態を1バイト長で返します。該当ビットが1か0かを判定することで状態を把握できます。 0000000x DTR 0=Low; 1=Hihg 000000x0 RTS 0=Low; 1=Hihg 00000x00 DSR 0=Low; 1=Hihg 0000x000 CTS 0=Low; 1=Hihg 000x0000 Overrun 0=NoErr; 1=OverrunErr 00x00000 Parity 0=NoErr; 1=ParityErr 0x000000 Flaming 0=NoErr; 1=FlamingErr x0000000 DLC 0=NoDLC 1=DLC is avalable

↓次に続きます

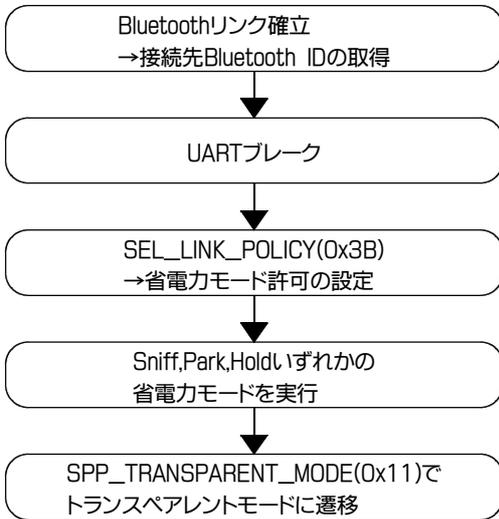
	3バイト目~4バイト目: 遅延時間(単位ms) (長さ2バイト) 0~3000までの範囲で空間伝搬遅延の時間を通知します。200ms単位です。0が返れば遅延がないことを示しています。

【4】*Bluetooth(R)*の低電力モード設定

*Bluetooth(R)*では、Sniffモード、Holdモード、Parkモードという3種類の低電力モードを用意しています。いずれのモードも、ホストへの応答回数を減らすなどの工夫で受信回路の休止時間を作り出し、消費電力を低減させようとするものです。

RBT-001では3つのモードをサポートとしていますが、いずれのモードにしても消費電力低減に大きな差はありません。最も使いやすいSniffモードにすると非Sniffモードより約35%程度の電力低減が期待できます。

RBT-001で使用する場合には、*Bluetooth(R)*リンク確立後に、UARTブレークを実行し、その状態でリンクポリシーを設定します。リンクポリシーは、現在のリンクに対するポリシーの設定で、どの省電力モードの使用を許可するかということを設定します。（すなわちリンクポリシーで設定できる省電力モードは常に1つだけです。またリンクポリシーを設定する場合には、接続先の*Bluetooth(R)*機器の*Bluetooth(R)* IDが必要です。）

リンクポリシー設定後に、Sniffモード又はParkモード、必要に応じてHoldモードを設定します。それぞれパラメーターがあるので設定には注意が必要です。下記にフローを示します。

リンクポリシーでのモード移行許可の設定は、省電力モード実行時には毎回設定する必要があります。省電力モードを実行した後は、一般的に"SPP_TRANSPARENT_MODE"コマンドを発行してトランスペアレントモードに移行します。

①Sniffモード

Sniffモードは、*Bluetooth(R)*の接続を維持したまま、一定時間データが送られてこなければ低消費電力モードに移行するモードです。

Sniffモードでは、設定したSniff周期毎の待機スロット分だけを受信します。よって、待機時の処理が減るため、受信回路の消費電力を抑えることができます。Sniff周期はRBT-001側で設定しても当然マスタ側にも通知されるため、マスタがSniffモードのRBT-001へパケットを送りたい場合にはSniff周期に合わせて送信してきます。

②Parkモード

Parkモードでは、一定のパーク周期を設定して、スレーブ機器はマスタ側からビーコンチャネルを受信します。（これをビーコン周期と言います）これにより、休止状態を作り出してもマスタとスレーブは同期しているため、Parkモードを終了した後に再度接続を継続できます。
Parkモードは接続を維持しながら比較的大きな省電力をできることが特徴ですが、Sniffモードと異なり、データ通信ができなくなります。

③Holdモード

Holdモードでは、ピコネットに同期した状態で、設定した期間中（ホールド時間中）は送受信を行わないことで省電力を実現します。ホールド時間経過後から通信を再開します。但しホールド期間は実時間で最大2.56秒までしか指定できません。2.56秒経過後は通常状態に戻りますので、省電力効果は持続的でないことに注意する必要があります。

リンクポリシーの設定	
詳細	指定した*Bluetooth(R)* ID機器とのリンクについてのポリシーを設定します。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_SET_LINK_POLICY(0x3B)
データ長	8
データ	1バイト目～6バイト目:　*Bluetooth(R)* IDの指定 (長さ6バイト) 現在接続している接続先デバイスの*Bluetooth(R)* IDを指定します。
	7バイト目以降:　許可するポリシーの設定（2バイト長） 0x0002　Holdモード許可 0x0004　Sniffモード許可 0x0008　Parkモード許可 データ長は2バイト長ですが、値が小さいので実際には1バイトしか使用しません。下位バイトを先に指定します。下記の例を参照下さい。
例	*Bluetooth(R)* IDが 00,00,00,00,00,00 のリモートデバイスとのリンクポリシーを"Sniffモード許可"に設定する場合 02,52,3B,08,00,95,00,00,00,00,00,00,04,00,03 *Bluetooth* ID　↓ 0x0004に設定するため下位バイトから先に指定 ※00,04としないようにご注意下さい。
戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト 1バイト目:　エラー通知 ERROR_OK（エラーなし）　0x00 予期しないエラーです　0x27 パラメータの指定が不正です　0x01 リンクが確立されていません　0x1F リンクポリシーの設定が不正です　0x2D
補足	エラーコード0x1Fが返る場合は、指定した*Bluetooth(R)* ID機器とのリンクが確立していない状態で、リンクポリシーを設定しようとしているためです。*Bluetooth(R)* IDの指定の間違いがないか、また本コマンドの発行を、リンク確立後のUARTブレーク中行っているか確認下さい。

リンクポリシーの設定を行うと、指定したリモートデバイスとの間で、省電力モードの設定ができるようになります。

Sniffモードを実行する

詳細	指定したSniffインターバルでSniffモードを実行します。 Sniffインターバルは、最大値と最小値を指定します。 また試行回数とタイムアウトをスロット数で指定します。 指定する値は正確に値を算出して行う必要があります。設定の詳細は、設定例をご覧下さい。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_ENTER_SNIFF_MODE(0x21)
ﾃﾞｰﾀ長	14
データ	1バイト目～6バイト目: *Bluetooth(R)* IDの指定 (長さ6バイト) Sniffモードを実行するリモートデバイスの*Bluetooth(R)* IDを指定します。このIDは先のリンクポリシーで設定したIDと同じ値にする必要があります。
	7バイト目: Sniffインターバルの最大値 (長さ2バイト) 2つ*Bluetooth(R)*機器間でパケットを交換しないスロットのインターバルの最大値を指定します。 範囲は0x0006～0x1000で、実時間では3.725ミリ秒～2.56秒となります。 ※計算例は設定例をご覧下さい。
	9バイト目: Sniffインターバルの最小値 (長さ2バイト) 2つ*Bluetooth(R)*機器間でパケットを交換しないスロットのインターバルの最小値を指定します。 範囲は0x0006～0x1000で、実時間では3.725ミリ秒～2.56秒となります。 ※計算例は設定例をご覧下さい。
	11バイト目: Sniff Attempt (長さ2バイト) Sniff Attempは、Sniffスロットが開始されて、スレーブがマスタからの発信パケットに応答するスロットの数のことです。 範囲は0x0001～0x7FFFまでです。 RBT-001では5スロットにすることが実験的に推奨されています。
	13バイト目: Sniffタイムアウト (長さ2バイト) タイムアウトするスロットの数を指定します。 範囲は0x0000～0x0028までです。RBT-001では3スロットにすることが実験的に推奨されています。
設定例	sniffモードでは、設定によって与えられた条件を元にしてマスターとスレーブが演算を行った後、Sniffモードに入ります。よって演算した結果パラメーターの値が不正の場合sniffモードを実行できません。必ず戻り値を確認して、snifモードに遷移できたかどうかをプログラム側で確認する必要があります。 ここでは設定例として、1パケット毎のインターバル間隔を最大300ms、最小50msに設定し、通信回数を低減させる設定にしてみます。 ※このインターバル間隔とは2つのデバイス間でデータのやりとりをしていない時の間隔です。 ①sniffインターバルの計算 インターバル間隔は4096ステップで設定でき、実時間にすると最大2.56秒です。よって1回のスロットは 2.56÷4096=625μs　となります。 最大300ms間隔に設定しますので、スロット数は 300ms÷625μs =480スロット となります。これを16進数に変換すると　0x01E0 となります。 よって最大Sniffインターバルは0x01E0スロットになります。 続いて同様にして最小値となる50msの場合も計算します。50ms÷625μs = 80スロット　となります。これを16進数に変換すると、0x0050 となります。 よって最小Sniffインターバルは0x0050スロットになります。 ②Sniff Attempとタイムアウトの値 上記のスロット数は、2デバイス間でデータのやりとりをしていない場合です。続いて実際にデータが到来した時の値をAteemptとタイムアウトで設定します。 ここではマスターからのデータ応答に対し5スロットを割り当てます。さらにデータが到来した場合はさらに3スロットを割り当てます。 Sniff Attempt = 5スロット Sniff タイムアウト = 3スロット ※上記のSniff Attempとタイムアウトは実測値より最も推奨される値です。 ③実際のコマンド設定 ①②の値よりコマンドフレームを作りRBT-001に送信します。ここではリモートデバイスの*Bluetooth(R)* IDが、00,00,00,00,00,00の場合を例に記載しています。 02,52,21,0E,00,81,00,00,00,00,00,00, →続く *Bluetooth(R)* ID E0,01,50,00,05,00,03,00,03 1E0　50　5　3 上記の設定にすると消費電流は平均10mA程度になります。(非sniffモード時平均17mA程度) なお、設定値を変更しても*Bluetooth(R)*の仕様で消費電流はほとんど変化しません。
補足	Sniffモードに設定したまま、コマンドでトランスペアレントモードに移行すれば(SPP_TRANSPARENT_MODE)、そのまま低消費電流の状態で、トランスペアレントモードで通信が可能になります。 Sniffモードから非Sniffモードに遷移するためには、"GAP_EXIT_SNIFF_MODE"のコマンドが必要です。Sniffモードでトランスペアレントモード中の場合には、一度UARTブレークしてから"GAP_EXIT_SNIFF_MODE"を発行して、非Sniffモードに遷移します。

↓　次のページに続きます

戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 予期しないエラーです　0x27 パラメータの指定が不正です　0x01 リンクが確立されていません　0x1F
補足	エラーコードが返る場合は次の点を確認してください。 ・モード設定前にリンクポリシーが設定してあるか ・*Bluetooth(R)* IDの設定を間違っていないか ・設定値の上位バイトと下位バイトを入れ違えていないか ・設定値に矛盾がないか

Sniffモードを終了する	
詳細	Sniffモードを終了して通常モードに移行します。
パケット	REQ(0x52)
オペコード	GAP_EXIT_SNIFF_MODE(0x37)
データ長	6
データ	1バイト目〜6バイト目:　*Bluetooth(R)* IDの指定 (長さ6バイト) Sniffモードを実行していた通信相手先のリモートデバイスの*Bluetooth(R)* IDを指定します。
戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 予期しないエラーです　0x27 パラメータの指定が不正です　0x01 リンクが確立されていません　0x1F

Holdモードを実行する	
詳細	Holdモードは指定したタイムレンジで、2デバイス間のパケット交換を停止し、省電力を実現します。 指定した時間が経過するとノーマル転送状態に戻ります。 指定できる実時間は最大2.56秒と短いため注意が必要です。 Holdインターバルは、最大値と最小値を指定します。 また試行回数とタイムアウトをスロット数で指定します。 指定する値は正確に値を算出して行う必要があります。設定の詳細は、設定例をご覧下さい。
パケット	REQ(0x52)
オペコード	GAP_ENTER_HOLD_MODE(0x3A)
データ長	10
データ	1バイト目〜6バイト目:　*Bluetooth(R)* IDの指定 (長さ6バイト) Holdモードを実行するリモートデバイスの*Bluetooth(R)* IDを指定します。このIDは先のリンクポリシーで設定したIDと同じ値にする必要があります。
	7バイト目: Holdインターバルの最大値　(長さ2バイト) 2つのデバイス間でデータ交換をしない時間分のスロット数の最大値を設定します。 範囲は0x0006〜0x1000で、実時間では3.725ミリ秒〜2.56秒となります。 ※計算例は設定例をご覧下さい。
	9バイト目: Holdインターバルの最小値　(長さ2バイト) 2つのデバイス間でデータ交換をしない時間分のスロット数の最小値を設定します。 範囲は0x0006〜0x1000で、実時間では3.725ミリ秒〜2.56秒となります。 ※計算例は設定例をご覧下さい。
設定例	Holdモードでは、設定によって与えられた条件を元にしてマスターとスレーブが演算を行った後、Holdモードに入ります。 ここでは設定例として、ホールドする時間の最大値を2000ms(2秒)、最小1000ms(1秒)に設定してみます。計算方法は先のSniffモードと同じです。 1スロットは625μsですので、2000msでは3200スロットとなります。3200スロットは0x0C80です。 続いて1000msは1600スロットで、0x0640です。 上記の設定値を、UARTブレーク中に送信します。 上記の設定にするとHoldモード期間中の消費電流は平均9.5mA程度になります。
補足	Holdモードに入るとすぐに"GAP_POWER_SAVE_MODE_CHANGED"(0x3D)のインジケーターが戻りモードが遷移したことを通知します。Holdモード期間が終了して通常状態へ遷移した時も同様に"GAP_POWER_SAVE_MODE_CHANGED"が返ります。

Parkモードを実行する	
詳細	Parkモードは、指定した比較的長い間隔でマスタからのビーコンチャネルを受信して同期を取りながら省電力モードへ移行するモードです。 Parkモード実行後は、直ちにトランスペアレントモードに移行してください。 非パークモード時と比較して約40%程度の省電力を期待できます。(平気約9.8mA程度の消費電流となります) ただしRBT-001側からParkモード時にデータを送信しようとすると大幅な遅延が発生します(約10秒程度)。また場合によってはバークモードは解除されます。一般的には送信するタイミングはUART機器側で分かりますので、送信するデータが発生した場合には、再度UARTブレークを実行して"GAP_EXIT_PARK_MODE"を発行してParkモードを脱してから、データを送信するようにします。 ホスト側からデータが到来した場合には、データの取りこぼしを防ぐため自動的にParkモードは終了し、通常モードに移行します。このような規則がありますので使用に際してはご注意下さい。 ビーコン間隔は最大1000ms、最小500msと設定するのが実験的に推奨されています。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_ENTER_PARK_MODE(0x38)
ﾃﾞｰﾀ長	10
データ	1バイト目～6バイト目: *Bluetooth(R)* IDの指定 (長さ6バイト) Parkモードを実行するリモートデバイスの*Bluetooth(R)* IDを指定します。このIDは先のリンクポリシーで設定したIDと同じ値にする必要があります。
	7バイト目: ビーコン間隔の最大値 (長さ2バイト) ビーコンチャネルをマスタが送信する間隔の最大値を実時間で指定します。 範囲は0x0006～0x1000で、実時間では3.725ミリ秒～2.56秒となります。 推奨値は、1000msです。 ※計算例は設定例をご覧下さい。
	9バイト目: ビーコン間隔の最小値 (長さ2バイト) ビーコンチャネルをマスタが送信する間隔の最小値を実時間で指定します。 範囲は0x0006～0x1000で、実時間では3.725ミリ秒～2.56秒となります。 推奨値は、500msです。 ※計算例は設定例をご覧下さい。
設定例	マスタが送信する同期確立維持のためのビーコンチャネルの間隔を実時間で指定します。この間隔が長すぎるとリンクが復帰できない場合があります。 実験的には最大値1000ms、最小値500msとすることが推奨されます。 1000ms→0x03E8　500ms→0x01F4
補足	Parkモードに入るとすぐに"GAP_POWER_SAVE_MODE_CHANGED"(0x3D)のインジケーターが戻りモードが遷移したことを通知します。 Parkモード実行後は、すぐに"SPP_TRANSPARENT_MODE"を発行してトランスペアレントモードに移行してください。移行しないとParkモードが周期的に無効となり省電力効果が少なくなります。 RBT-001側からマスタ側にデータ送信する時は、遅延発生を防ぐため、UARTブレークしたのち、"GAP_EXIT_PARK_MODE"でParkモードを終了させてから、"SPP_TRANSPARENT_MODE"を発行してトランスペアレントモードに移行しデータを送信することをお奨めします。 マスタからデータが到来した場合には自動的にParkモードは終了し通常の電力モードに移行します。
戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 予期しないエラーです　0x27 パラメータの指定が不正です　0x01 リンクが確立されていません　0x1F
補足	エラーコードが返る場合は次の点を確認してください。 ・モード設定前にリンクポリシーが設定してあるか ・*Bluetooth(R)* IDの設定を間違っていないか ・設定値の上位バイトと下位バイトを入れ違えていないか ・設定値に矛盾がないか

Parkモードを終了する	
詳細	Parkモードを終了して通常モードに移行します。
ﾊﾟｹｯﾄ	REQ(0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_EXIT_PARK_MODE(0x39)
ﾃﾞｰﾀ長	6
データ	1バイト目～6バイト目: *Bluetooth(R)* IDの指定 (長さ6バイト) Parkモードを実行していた通信相手先のリモートデバイスの*Bluetooth(R)* IDを指定します。
戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 予期しないエラーです　0x27 パラメータの指定が不正です　0x01 リンクが確立されていません　0x1F

パワーモード変更の通知	
詳細	RBT−001のパワーモードが変更されると通知されます。
ﾊﾟｹｯﾄ	IND(0x69)
ｵﾍﾟｺｰﾄﾞ	GAP_POWER_SAVE_MODE_CHANGED(0x3D)
ﾃﾞｰﾀ長	8
データ	1バイト目:　ステータス情報　(長さ1バイト) ERROR_OK(エラーなし)　0x00 試行エラーです(不正なﾊﾟﾗﾒｰﾀｰ)　0x2C 予期しないエラーです　0x27
	2バイト目〜7バイト目:　リモート*Bluetooth(R)* ID (長さ6バイト) パワーモードが変更されたリモート*Bluetooth(R)*のBluetooth IDが6バイトで返ります。
	8バイト目:　変更されたモード（長さ1バイト） アクティブモード(通常モード)　0x00 Holdモードが実行されました　0x01 Sniffモードが実行されました　0x02 Parkモードが実行されました　0x03
補足	パワーモードが変更された場合には、必ずこのGAP_POWER_SAVE_MODE_CHANGEDが返ります。但し、トランスペアレントモード移行後にパワーモードが変更された場合(Parkモード実行中にホストからのデータを受信して自動的にParkモードが終了した場合など)には、何も返りません。

【5】RBT−001の設定

RBT−001本体の設定を行うコマンドです。
*Bluetooth(R)*機器は、*Bluetooth(R)* IDという固有のIDでそれぞれの機器を認識していますが、このIDは数字の羅列で人間には直感的に分かりにくいためこれとは別にデバイスネームを付けることができます。本設定では、こういったローカルデバイスの設定を行うコマンドを紹介します。

自機のデバイスネームの取得	
詳細	自機に設定されているデバイスネームを取得します。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_READ_LOCAL_NAME(0x03)
ﾃﾞｰﾀ長	0

自機のデバイスネームの通知	
詳細	自機に設定されているデバイスネームを通知します。
ﾊﾟｹｯﾄ	CFM (0x43)
ｵﾍﾟｺｰﾄﾞ	GAP_READ_LOCAL_NAME(0x03)
ﾃﾞｰﾀ長	2+ネームの長さ
データ	1バイト目;　ステータス情報　(長さ1バイト) ERROR_OK(エラーなし)　0x00 パラメーターの誤りです　0x01 予期しないエラーです　0x05
	2バイト目:　デバイスネームの長さを表す値(長さ1ﾊﾞｲﾄ) 最大40バイトです。 ※文字列の終端のNULL1バイト分を含みます
	3バイト目以降:　デバイスネームが返ります。 文字列はNULLで終端されます。最大40バイトです。
例	デバイスネームが"Serial Port Device"の場合 02,43,03,15,00,5B,00,13, デバイスネームの文字数 53,65,72,69,61,6C,20,50,6F,72,74,20,44,65,76,69,63,65,00,03 ↓ デバイス名

自機のデバイスネームの変更	
詳細	自機のデバイスネームを任意の名前に変更します。 最大40バイトまで指定できますが、文字列の最後は必ずNULLで終端します。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	GAP_WRITE_LOCAL_NAME(0x04)
ﾃﾞｰﾀ長	1+デバイスネームの長さ
データ	1バイト目:　デバイスネームの長さ(長さ1バイト) 新たに付けるデバイス名の長さ(バイト数)を指定します。名前の最後にはNULL(0x00)を付加しますので、デバイス名+1バイトで指定します。
	2バイト目以降:　設定するデバイスネーム 新たに設定するデバイスネームを40バイト以内で指定します。最後は必ずNULL(0x00)で終端してください。
例	デバイスネームを"TEST"と設定する場合 02,52,04,06,00,5C,05,54,45,53,54,00,03 "TEST"+0x00 ※指定するデバイスネームの長さは実際の文字列+NULL1バイト分となっていることに注意して下さい。
戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 パラメータの指定が不正です　0x01 デバイス名が長すぎます　0x06

イベントフィルターの設定	
詳細	イベントフィルターとは、イベントが発生した時にどのように対応するかについての設定のことです。 UARTブレークの有効/無効も設定できます。 イベントが発生した時の処理を設定します。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	SET_EVENT_FILTER(0x4E)
ﾃﾞｰﾀ長	1
データ	1バイト目:　設定値　(長さ1バイト) 0x00 すべてのイベント報告を有効 0x01 ACLリンクインジケーターのみ無効(ﾃﾞﾌｫﾙﾄ設定) 0x02 イベントレポートを無効、UARTブレークは有効 0x03 イベントレポートを無効、UARTブレークは無効 RS232Cの有線通信のエミュレートモード
補足	イベントレポートは、RBT-001が返すコンファメーション(CFM)及びインジケーターを(IND)のことを言います。イベントレポートを無効に設定すると、リクエストに対するコンファメーションが戻らなくなります。コンファメーションが必要ないときは、イベントレポートを無効にすることで通信量を減らすことができます。 また、引数に0x03を指定するとイベントレポートが無効になり、さらにUARTブレークも無効になります。この状態は2デバイス間を有線でRS232C通信した時に最も状態が近いエミュレーション状態となります。 なお、工場出荷時のデフォルト設定は、0x01となっています。この設定では、ACLに関するインジケーターだけを無効にしています。
戻り値	パケット・・・　CFM(0x43) データ長・・・　1バイト 1バイト目:　エラー通知 ERROR_OK(エラーなし)　0x00 パラメータの指定が不正です　0x01 ※0x02及び0x03に設定した場合、戻り値は戻りません。

RBT－001の設定を工場出荷時の設定に戻す	
詳細	RBT－001の不揮発性メモリーの内容を工場出荷時の状態に戻します。ユーザーが設定した内容はすべて破棄されて工場出荷時の設定状態になります。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	RESTORE_FACTORY_SETTING(0x1A)
ﾃﾞｰﾀ長	0
戻り値	パケット・・・ CFM(0x43) データ長・・・ 1バイト 1バイト目: エラー通知 ERROR_OK(エラーなし) 0x00 パラメータの指定が不正です 0x01
補足	工場出荷時の状態に戻すと、リンク先の*Bluetooth(R)*機器とは再度ペアリングの作業が必要となります。 パスキーのデフォルト値は0000となっています。

本体をリセットする(再起動)	
詳細	RBT－001をリセットします。
ﾊﾟｹｯﾄ	REQ (0x52)
ｵﾍﾟｺｰﾄﾞ	RESET(0x26)
ﾃﾞｰﾀ長	0
戻り値	リセット後は"DEVICE READY"が戻ります。

デバイスの起動を通知します(DEVICE READY)	
詳細	RBT－001が起動したことを通知します。 デバイス起動時は、このメッセージを受信後から各種操作を始めるようにします。 ※イベントレポートを無効にしている場合には、この値は返りませんのでご注意下さい。
ﾊﾟｹｯﾄ	IND (0x69)
ｵﾍﾟｺｰﾄﾞ	DEVICE_READY (0x25)
ﾃﾞｰﾀ長	1+バージョン値の長さ
戻り値	1バイト目: バージョンの文字列の長さ(長さ1ﾊﾞｲﾄ) バージョン値の文字列の長さを返します。
	2バイト目以降: 本体のファームウエアバージョン 本体のファームウエアバージョンを返します。

【6】ACLインジケーター

ACLはAsynchronous ConnectionLessで非同期接続のことを示します。この非同期というのは、データの受信と送信が非同期、すなわち1対1となっていない通信のことです。ACLでは1スロット分のパケットをホストがスレーブに対して送信してもスレーブはそれに同期せず例えば3スロット分のデータを送信してもよい接続方法のことをいいます。スロット数が1対1ではないので、通信速度は均一化しませんが非対称性な通信が可能です。

RBT－001ではACLで通信を行います。ACLインジケーターは、ACLのリンクが確立した場合と、リンクが終結した場合の2つのパターンで通知を行います。

なお、"SET_EVENT_FILTER"で"Report all events"を設定しておかないとACLインジケーターは通知されません。

ACLリンク確立の通知	
詳細	RBT－001がリモートデバイスとACLリンクを確立したことを通知します。
ﾊﾟｹｯﾄ	REQ (0x69)
ｵﾍﾟｺｰﾄﾞ	GAP_ACL_ESTABLISHED (0x50)
ﾃﾞｰﾀ長	7
戻り値	1バイト～6バイト目; リモートデバイスの*Bluetooth(R)* ID (長さ6バイト)
	7バイト目: ステータス情報 (長さ1バイト) 0x00の時のみエラーがないことを通知します。 その他の値については本書に掲載の"ACLエラーリスト"をご参照下さい。

ACLリンク終結の通知	
詳細	リモートデバイスとのACLリンクがターミネート(終結)したことを通知します。
ﾊﾟｹｯﾄ	REQ (0x69)
ｵﾍﾟｺｰﾄﾞ	GAP_ACL_TERMINATED (0x51)
ﾃﾞｰﾀ長	7
戻り値	1バイト～6バイト目; リモートデバイスの*Bluetooth(R)* ID (長さ6バイト)
	7バイト目: ステータス情報 (長さ1バイト) 0x00の時のみエラーがないことを通知します。 その他の値については本書に掲載の"ACLエラーリスト"をご参照下さい。

ユーティリティーソフトウエアを使用する

RBT-001のUARTコマンドをより簡単にパソコンから送信できる"SimpleBlueCommander for RBT-001"を配布しております。このソフトウエアは、ナショナルセミコンダクター社のリリースしている*Bluetooth(R)*デバイス用のソフトウエア"SimpleBlueCommander"に、RBT-001用のコマンド定義ファイルを追加したソフトウエアです。

RBT-001のUART信号線をパソコンのRS232Cポートに接続している場合には、このソフトウエアから各種シリアルコマンドを送信することができます。またRBT-001が受信したデータをウインドウに表示します。各コマンドは本ソフトウエア内で解析されて値だけでなく、コマンド名に変換して表示することもできます。面倒なチェックサムの計算もできますので、シリアルターミナルとしてだけでなく、シリアルコマンドのフォーマット作成にも利用できます。

※パソコンのRS232CポートとRBT-001を接続する場合には、必ずレベル変換ICを介してロジック信号の電圧が3Vp-pの振幅になるよう設計をしてください。(本書3ページに参考回路があります。)

■SimpleBlueCommanderをインストールする

SimpleBlueCommanderは、当方のwebサイトからダウンロードすることができます。下記サイトよりダウンロードしてください。

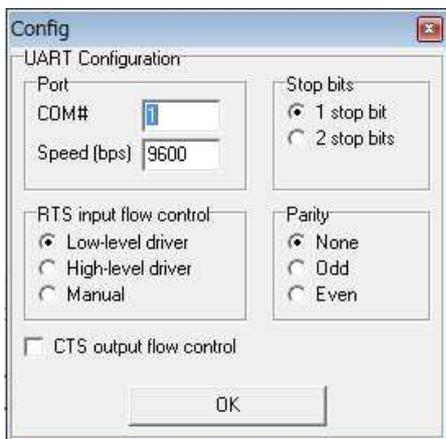
http://www.microtechnica.tv/support/software/sbc.zip

ZIP形式で圧縮されています。解凍すると、ファイルが展開されます。インストーラーは付属していませんので、適当なディレクトリに配置してご使用下さい。本体ファイルは"SimplyBlueCommander.exe"ですので、適宜デスクトップにショートカットなどを作成されると便利です。

■SimpleBlueCommanderの設定

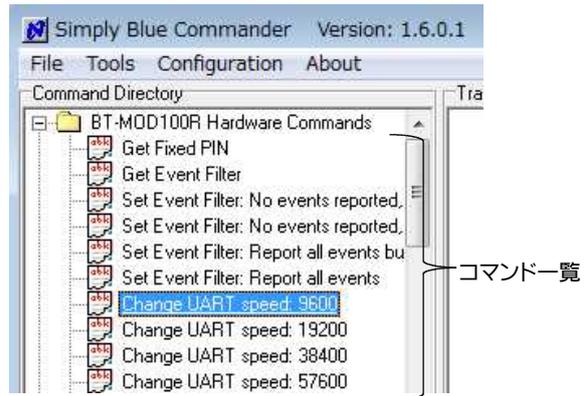
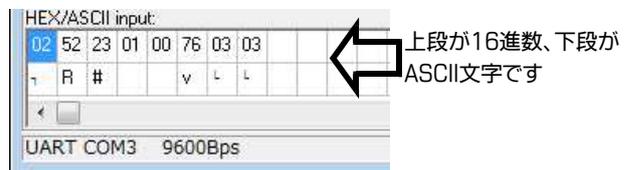
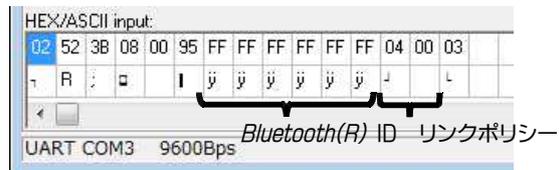
SimpleBlueCommanderを最初に起動した時、"Cannot open UART"と表示されることがあります。これは、RS232Cポートの設定をしていないためですので、そのままOKボタンを押して続行します。

プログラムが実行されますので、最初にRBT-001と接続しているCOMポートの設定を行います。RBT-001をパソコンに接続していない場合には、設定は必要ありません。(但し毎回起動時に"Cannot open UART"のメッセージが表示されます。)

メニューバーの"Configuration"をクリックして、"Transport Layer"をクリックします。下図のようなウインドウが表示されます。

"COM#"のテキストボックスにRBT-001を接続したCOMポートの番号を入力します。"Speed"は、RBT-001のUART通信速度をbps単位で設定します。工場出荷時の設定は9600bpsとなっています。その他の設定は上記の通りに設定します。

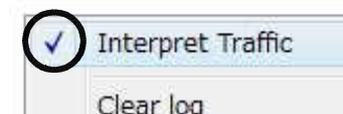
設定が完了したらOKボタンを押して閉じます。

■SimpleBlueCommanderの基本的な使い方

よく使用するコマンドは、ウインドウ左側の"Command Directory"に登録されています。フォルダアイコンの左側の+印をクリックすると、コマンド一覧が表示されます。

コマンドをクリックすると、ウインドウ下部の"HEX/ASCII input"にコマンドの値が16進数とASCII文字で表示されます。

コマンドによっては、*Bluetooth(R)* IDを入力したり、設定値を入力しなければならない場合があります。特に*Bluetooth(R)* IDはすべて"FF"で表示されますので、お客様の環境に合わせて入力してください。

上図は、"Set Link Policy"をクリックした時の例です。FFの部分にはリモート*Bluetooth(R)*デバイスの*Bluetooth(R)* IDを入力します。また、上図では、リンクポリシーの設定2バイトの部分が0x0004に設定されています(下位バイト先頭)。リンクポリシー0x0004はSniffモードの許可ですので、Parkモード又はHoldモードの許可をしたい場合には、ここの値を適宜正しい値に変更する必要があります。

コマンドの内容が正しいことを確認したら、"Send"ボタンを押します。コマンドが指定されたCOMポートより出力されます。出力された値と、RBT-001から返った値は"Transport Layer log"のウインドウに表示されます。下図は、"Set Link Policy"のコマンドを送信した例です。

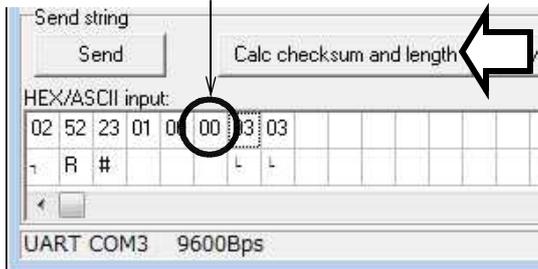
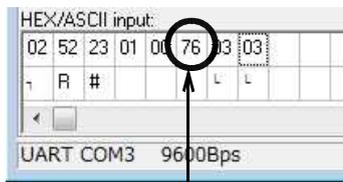
送信はTx: で、受信したデータはRx: で表示されます。

なお、コマンドを解析して表示するのではなく、16進数の値をそのまま表示させたい場合には、"Transport Layer log"のウインドウ内を右クリックし、メニュー内の"Interpret Traffic"のチェックを外してください。

■SimpleBlueCommanderでUARTブレーク信号を出力する

UARTブレーク信号を出力したい場合には、"Generate break"ボタンをクリックします。約30msの0x00が出力され、RBT-001はUARTブレークモードになります。

■コマンドを手動で入力する方法、チェックサムの計算方法

あらかじめ登録されているコマンドではなく、手動でコマンドを入力する場合には、"HEX/ASCII input"の部分に16進数の値又は文字を入力します。1マスが1バイト分に相当します。1マスに入力をし終えると自動的に次のマスにカーソルが移動します。

チェックサムの値は、ソフトウエアで計算して入力させることが可能です。その場合、チェックサムの入るべきマスには必ず "00" を入力して最初にすべてのデータを入力しておきます。例えば下図は、UARTの通信速度を9600bpsに設定する例です。チェックサムの入るべき6バイト目には 00 を入力してその他の値を入力しておきます。

エンドデリミタの 03 まで入力が完了したら、"Calc chacksum and length"ボタンをクリックします。チェックサムの値が計算されて入力されます。

計算されて入力されます。

チェックサムの計算は面倒なので、このソフトウエアを使用すると簡単に値を算出できます。RBT-001をパソコンと接続されていない場合でも、コマンドフォーマットを作る際にお役立て頂けます。

実際の使用例

ここでは、RBT-001を実際に使用した場合の例を項目別に掲載します。送信したシリアルコマンドの値、またRBT-001から戻った値などを記載しています。実際の通信を行う場合の参考としてお役立て下さい。なお、RBT-001の設定は工場出荷時の設定とした場合です。

■リモートデバイスとのペアリングを行う

※"SET_EVENT_FILTER"の設定が工場出荷時のデフォルト設定になっているとACL関連は通知されません。下記は、"SET_EVENT_FILTER"の設定を、"Report all events"に設定した場合です。

```
02,69,50,07,00,C0,FF,FF,FF,FF,FF,FF,FF,00,03
02,69,51,07,00,C1,FF,FF,FF,FF,FF,FF,FF,13,03
```

最初にACLのリンク確立"GAP_ACL_ESTABLISHED"が通知されます。データ長は7バイトでリモートデバイスの*Bluetooth(R)* IDが含まれます。(上記ではFFで表記しています。)
続いて、ACLリンクは一度ターミネートされます。ACLターミネートの理由は、"User ended connection"です。

※以下の例ではすべてACLの通知は無効にしているものとします。

■RBT-001電源投入後に、*Bluetooth(R)*リンクを確立する

```
02,69,25,05,00,93,04,30,32,31,32,03     ①
02,69,0C,07,00,7C,FF,FF,FF,FF,FF,FF,01,03     ②
```

①RBT-001が起動し、準備が完了したことを通知します。
②*Bluetooth(R)*リンクを確立に成功するとリモートデバイスの*Bluetooth(R)* ID6バイト(上記FF)と、RFCOMM番号が返ります。

②のメッセージ受信後からトランスペアレントモードが実行されていますので、以後のデータはすべて*Bluetooth(R)*でリモートデバイスとの間で送受信されます。

■*Bluetooth(R)*リンクを切断された場合

```
00     ①
02,69,11,02,00,7C,01,00,03     ②
02,69,0E,02,00,79,01,01,03     ③
```

①最初の"00"は、*Bluetooth(R)*リンクが切断された時、約6.8ms程度の0信号を表しています。UARTとしてのデータはないですが、ターミナルソフトによっては 00 と判定することがあります。下図はリンク切断直後のRBT-001のTXピンの波形です。
"02"が到達する前に電位レベルが0Vの状態が約6.8ms出力されていることが確認できます。

②トランスペアレントモードが終了するため、"SPP_TRANSPARENT_MODE"の0x11が返ります。
③続いて"SPP_LINK_RELEASED"が返ります。

RBT-001のオペコード一覧とACLエラーコード

■オペコード一覧

オペコードの内容	値
GAP_READ_LOCAL_NAME	0x03
GAP_WRITE_LOCAL_NAME	0x04
GAP_READ_LOCAL_BDA	0x05
GAP_GET_FIXED_PIN	0x16
GAP_SET_FIXED_PIN	0x17
GAP_REMOVE_PAIRING	0x1B
GAP_LIST_PAIRED_DEVICE	0x1C
GAP_ENTER_SNIFF_MODE	0x21
GAP_EXIT_SNIFF_MODE	0x37
GAP_ENTER_PARK_MODE	0x38
GAP_EXIT_PARK_MODE	0x39
GAP_ENTER_HOLD_MODE	0x3A
GAP_SET_LINK_POLICY	0x3B
GAP_GET_LINK_POLICY	0x3C
GAP_POWER_SAVE_MODE_CHANGED	0x3D
GAP_ACL_ESTABLISHED	0x50
GAP_ACL_TERMINATED	0x51
SPP_ESTABLISH_LINK	0x0A
SPP_INCOMMING_LINK_ESTABLISHED	0x0C
SPP_RELEASE_LINK	0x0D
SPP_LINK_RELEASED	0x0E
SPP_SEND_DATA	0x0F
SPP_INCOMING_DATA	0x10
SPP_TRANSPARENT_MODE	0x11
CHANGE_NVS_UART_SPEED	0x23
CHANGE_UART_SETTINGS	0x48
RESTORE_FACTORY_SETTINGS	0x1A
SET_EVENT_FILTER	0x4E
GET_EVENT_FILTER	0x4F
RESET	0x26
RBT-001 READY	0x26

上記のオペコード一覧は本書にて解説しているオペコードです。RBT-001にはこの他にも拡張用やテスト用にオペコードが用意されています。よってお客様が上記以外のオペコードを送信すると、RBT-001の動作に不具合が生じたり、期待した動作をしなくなってしまうことがあります。システムの設計や実際の使用に際しては、上記以外のオペコードは送信しないよう、ご注意下さい。

※新しいオペコードは将来追加されることがあります。その場合には当方のサポートページでご紹介する他、マニュアルも更新されますので、定期的に当方のFAQページをご覧下さい。(マニュアルの更新回数はマニュアル1ページ目に記載のリビジョン番号で確認できます。)

■ACLエラーコード一覧

0x01	Unknown HCI Command.
0x02	No Connection.
0x03	Hardware Failure.
0x04	Page Timeout.
0x05	Authentication Failure.
0x06	Key Missing.
0x07	Memory Full.
0x08	Connection Timeout.
0x09	Max Number Of Connections.
0x0A	Max Number Of SCO Connections To A Device.
0x0B	ACL connection already exists.
0x0C	Command Disallowed.
0x0D	Host Rejected due to limited resources.
0x0E	Host Rejected due to security reasons.
0x0F	Host Rejected due to remote device is only a personal device.
0x10	Host Timeout.
0x11	Unsupported Feature or Parameter Value.
0x12	Invalid HCI Command Parameters.
0x13	Other End Terminated Connection: User Ended Connection.
0x14	Other End Terminated Connection: Low Resources.
0x15	Other End Terminated Connection: About to Power Off.
0x16	Connection Terminated by Local Host.
0x17	Repeated Attempts.
0x18	Pairing Not Allowed.
0x19	Unknown LMP PDU.
0x1A	Unsupported Remote Feature.
0x1B	SCO Offset Rejected.
0x1C	SCO Interval Rejected.
0x1D	SCO Air Mode Rejected.
0x1E	Invalid LMP Parameters.
0x1F	Unspecified Error.
0x20	Unsupported LMP Parameter Value.
0x21	Role Change Not Allowed
0x22	LMP Response Timeout
0x23	LMP Error Transaction Collision
0x24	LMP PDU Not Allowed
0x25	Encryption Mode Not Acceptable
0x26	Unit Key Used
0x27	QoS is Not Supported
0x28	Instant Passed
0x29	Pairing with Unit Key Not Supported
0x2A	Different Transaction Collision
0x2B	Reserved
0x2C	QoS Unacceptable Parameter
0x2D	QoS Rejected
0x2E	Channel Classification Not Supported
0x2F	Insufficient Security
0x30	Parameter out of Mandatory Range
0x31	Reserved
0x32	Role Switch Pending
0x33	Reserved
0x34	Reserved Slot Violation
0x35	Role Switch Failed

RBT-001の色々な使い方

本項では、RBT－001の様々な使い方を紹介しています。RBT－001では、1台のパソコンと1台のRBT－001でシリアル通信を無線化することが基本ですが、色々と応用させて使用することができます。

■1対1で、RBT－001とパソコン間で通信をする場合

最も基本的な通信形態です。1台のRBT－001と、1台のパソコン間で*Bluetooth(R)*によるシリアル通信を行います。パソコン側には、仮想COMポートが作られ、パソコン側のアプリケーションはこの仮想COMポートへアクセスすることで、遠隔地にあるRBT－001に接続されたUART機器と通信ができます。

1　RBT－001に電源を給電します。UART側にマイコン等の機器を接続します。

2　*Bluetooth(R)*通信機能搭載のパソコンを起動し、コントロールパネル内の"*Bluetooth(R)*設定"からRBT－001を検索し、検出します。
※"EasyBT"として検出されます。

3　ペアリングを行います。パスキーに"0000"を指定します。
ペアリングが行われ、パソコン側には仮想COMポートが作られます。表示された"発信COMポート"の番号がパソコン上から使用するCOMポート番号となります。

4　ここまでの状態ですと、RBT－001はUARTブレークの状態でコマンド受付状態です。データのシリアル通信を行う場合には、トランスペアレントモードを実行する必要があります。
パソコン側から手順3で作られた発信COMポートをオープンすると自動的にトランスペアレントモードが実行されます。
※UARTブレークの状態でデータの送受信を行いたい場合には、SPP_SEND_DATA(0x0F)並びに、SPP_INCOMING_DATA(0x10)を使用します。

5　仮想COMポートを閉じると、トランスペアレントモードは切断されます。

■複数のRBT－001を1台のホストPCと接続する場合

RBT－001には、1台ずつユニークな固有の*Bluetooth(R)* IDが書き込まれています。よって、複数のRBT－001を1台のホストパソコンに*Bluetooth(R)*で接続することができます。

仕組みとしては、RBT－001ごとに仮想COMポートがホストパソコン側に作成されますので、それぞれの仮想COMポートにアクセスすることで、それぞれのRBT－001と通信ができます。

基本的にはペアリングをして、ホストパソコン側に仮想COMポートが作成された段階で使用できるようになります。

RBT－001の電源を投入すると、*Bluetooth(R)*搭載のパソコンでRBT－001を検出します。検出の方法は本書6ページの手順と同様ですので、6ページの手順に従い検出します。さらに別のRBT－001の電源を投入すると、上記と同様の手順で新しいRBT－001を検出することができますのでペアリングをします。ペアリングをするごとに仮想COMポートが割り当てられますので、どのRBT－001に何番の仮想COMポートが割り当てられたかをメモしておくことが重要です。コマンドで*Bluetooth(R)* IDを取得することで識別することもできます。

■RBT－001同士で通信を行う場合

RBT－001をパソコンとではなく、2台のRBT－001同士で通信させることができます。その場合には、手動でSPPのリンクを確立し、RFCOMMを両デバイス間でオープンする必要があります。

あらかじめ通信する相手のRBT－001の*Bluetooth(R)* IDを知っておく必要があります。GAP_READ_LOCAL_BDR(0x05)で調べておくか、又は通信可能範囲内に、動作中のRBT－001が存在していれば、GAP_INQUIRY(0x00)コマンドで検出することもできます。

特定のRBT－001間で通信を行う場合には、トランスペアレントモードを実行することで、シリアルデータのバースト転送ができます。複数のRBT－001と通信を行うこともできますが、その場合には、RFCOMMの割り当てがそれぞれで必要になります。

1　2台のRBT－001に電源を給電します。ここでは、自機をローカルデバイス、通信相手方のRBT－001をリモートデバイスと記載します。

2　ローカルデバイスに、SPP_ESTABLISH_LINK(0x0A)を送信して、リンクを確立させます。この時、接続するリモートデバイスのBluetooth IDを指定します。
RFCOMMポートの番号は通常1から順番に使います。例えば、ローカルRFCOMM、リモートRFCOMMを共に1として指定してリンクを確立させます。
(リモートデバイスの*Bluetooth* IDが00:17:A0:01:02:03の場合)
02,52,0A,08,00,64,01,03,02,01,A0,17,00,01,03　とします。
Bluetooth(R) ID
※*Bluetooth(R)* IDはリトルエンディアンで指定します。

3　接続が正しく確立されると、リモートデバイス側ではSPP_INCOMING_LINK_ESTABLISHED(0x0C)がINDモードで返ります。
この時接続を要求してきた側の*Bluetooth(R)* IDと、RFCOMMの番号が返ります。
ローカルデバイス側では、SPP_LINK_ESTABLISHED(0x0B)が返ります。また、RFCOMMのポート状態が変わると、SPP_PORT_STATUS_CHANGED(0x3E)が返ることがあります。

4　手順3で接続したリモートデバイス間でシリアルデータのバースト転送を行う場合には、設定したRFCOMMポートに対して、トランスペアレントモードを実行します。
SPP_TRANSPARENT_MODE(0x11)を発行します。この時、手順2で指定したローカルRFCOMM番号を指定します。
SPP_TRANSPARENT_MODEコマンド発行後から、シリアルデータのバースト転送が可能となります。

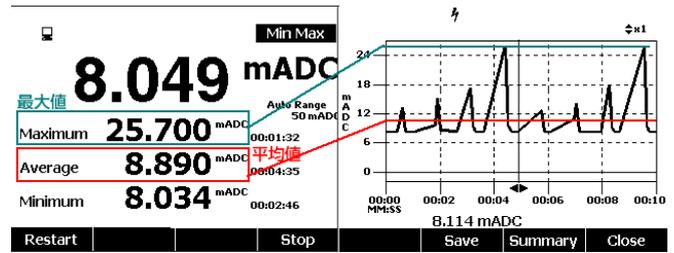
複数のRBT－001と通信を行う場合には、再度SPP_ESTABLISH_LINKコマンドを使用して、別のリモートデバイスの*Bluetooth(R)* IDを指定し、RFCOMMの番号を変えることで、複数のリモートデバイスと通信を確立できます。但し、複数のリモートデバイス間でシリアルデータの通信を行う場合には、トランスペアレントモードは使用できませんので(トランスペアレントモードは1対1の接続確立時のみ使用できます)、UARTブレーク中にデータを送受信できるコマンド、SPP_SEND_DATA(0x0F)コマンド並びに受信側は、SPP_INCOMING_DATA(0X10)を使用してUARTモードのパケット内でフレーミング通信を行います。

5　接続を切断したい場合には、トランスペアレントモード実行中の場合には、UARTブレークにして、ローカルデバイスをコマンド受信可能な状態にし、SPP_RELEASE_LINK(0x0D)コマンドを発行して、リンクを切断します。トランスペアレントモードを実行していない場合には、SPP_RELEASE_LINKを発行するだけで切断できます。

主な仕様

電源電圧:	DC3V(最小+2.5V～最大+3.3V) 絶対最大定格が+3.3V
消費電流: (下記参照) (Vcc=3.0V時)	SPPリンク未確立時　約9mA(Ave.) SPPリンク確立時　　約17mA(Ave.) SPPリンクデータ送受信時　約27mA(Ave.) UARTブレークモード中　約17mA(Ave.) *Bluetooth(R)*規格省電力モード実行時はパラメーターにより差はあるが、SPPリンク確立時は約35%程度の消費電流低減が期待できる
*Bluetooth(R)*準拠規格	*Bluetooth(R)* Ver.2.1 Class2 operation *Bluetooth(R)* Ver.1.xとの互換性有り
無線伝送方式	周波数ホッピング方式 スペクトラム拡散(FS-SS)方式 送信周波数範囲 2.4GHz～2.4835GHz
通信範囲	Class2準拠　見通し内通信で約30m程度
UARTインターフェイス	CTS/RTSハンドシェイク通信方式 但し、2線式通信でも通信可能 デフォルト設定通信速度　9600bps 通信速度最大921.6kbps
対応プロファイル	GAP, SDAP, SPP
動作環境:	5℃～60℃ (動作保証範囲) 但し結露無きこと
外形寸法	29mm×29mm
認証コード	技適認証　006NYC0046 *Bluetooth(R)* QPID　B0313300 *Bluetooth(R)* QDID　B011563
製造メーカー	RoboTech srl 本製品の技適認証並びに*Bluetooth(R)* SIGへの登録は同社によって行われています。 同社指定の型式はRBT-001です。
製造国	イタリア

※消費電流について

本製品はBluetoothの仕様上、定期的に本機の近傍にBluetooth通信機器がないかを検索しているめ、電波を発信しています。そのためリンク未接続時であっても消費電流は定期的に変動しており、多く流れる場合で25mA程度の電流が瞬間的に流れます。通常は8mA程度の消費電流であり、25mA程度の電流が流れるのは瞬間的ですので平均化すると、おおよそ消費電流は9mA程度となります。消費電流をグラフ化したものを次のページに示します。

なお当方で60分間、実機で計測した時の平均消費電流値は8.82mAでした。最大値は25.87mAでした。なおこれは本機の仕様であり異常ではありません。

また、SPPリンク確立時も同様であり、その場合平均消費電流は、約17mA程度で、最大24mA程度の電流が流れることがあります。瞬間的に比較的大きな電流が定期的に流れていることにご留意ください。(平均消費電流は瞬間的に流れる電流も含めて算出した値であり、定常時よりも若干高くなっています。)

使用上の注意

①RBT-001を使用するに際し、当方は明示的及び潜在的な使用したことによる危険性や、不確実性については予見することができません。使用する際には、お客様の責任においてこの製品を正しくお使いいただけますようお願い致します。

②RBT-001は、*Bluetooth(R)*規格に基づきデータ通信を空間伝送できるデータ通信モジュールですが、性能及び信頼性は一般的な使用の範囲に限定されます。本製品を宇宙、航空、原子力、燃焼制御、運輸、交通、各種安全装置、ライフサポート関連の医療機器等のように、特別な品質・信頼性を要求され、その故障や誤動作が直接人命を脅かしたり、人体に危害を及ぼす恐れのある用途向けには使用できません。

③RBT-001では様々な外的要因、特に使用する周囲の電磁界環境によって、性能を仕様どおり発揮できない場合があります。また、場合によっては正しく動作しない場合や期待した動作をしない場合があります。本製品を使用することによって生じた、もしくはこれに関連するいかなる直接・間接損害、懲罰的損害、その他データの破損や消失等を含むいかなる損害、損失についても、当方では一切責任を負いかねます。あらかじめご理解とご了承頂けますようお願い致します。

④RBT-001を使用した製品等を製造させる場合には、様々なフェイルセーフ機能(安全設計)を施して頂き、十分に機器のテストをした上で運用されますようお願い致します。また、データの損失や予期しない事態に備え、データのバックアップを行って頂けますようお願い致します。

⑤RBT-001は電波法に基づく無線設備として技術基準適合証明を受けています。本製品は日本国内でのみ使用することができます。海外での使用及び輸出はできません。また当方ではそれに関するご質問等についてはお答え致しかねます。RBT-001は日本国内でご使用下さい。また、RBT-001は技適認証をうけた製品ですので、本製品を分解・改造することは法律で禁止されています。また、基板に印刷されているマーク及び承認番号を消すことは禁止されています。

⑥RBT-001は以下の無線局と同じ周波数帯を使用します。
産業、科学、医療用機器・工場の製造ライン等で使用されている移動体識別用無線局・無線LAN関連製品全般上記の機器や無線局の近くで本製品を使用すると電波干渉が生じる恐れがあります。特に医療機器の近くでの使用は、他の機器に影響を与える場合がありますので使用に際しては十分ご注意下さい。

製品の技術的なサポートについて

本製品の技術的なサポートは製品の開発元、イタリアRoboTech Srl社が直接行います。技術的なサポートが必要な場合には、開発元へ直接ご連絡頂きます。原則と致しまして当方での技術的なサポートは致しておりません。
技術サポートはすべて英文となります。あらかじめご了承頂けますようお願い申し上げます。なお、日本語マニュアル及びFAQにつきましては、当方のサイトより最新の情報をご提供致します。

なお、サポートはいずれも技術者専門となっておりより技術的なご質問をするためのものとなっております。マニュアルに既に記載されている内容などについては回答が得られない場合がございます。

n.canelli@robotechsrl.com
info@robotechsrl.com

上記アドレスにご使用製品名(RBT-001)とご使用者様の会社名、学校名、お名前、メールアドレスをご記入頂き、メールをお送り下さい。
なおご質問事項はできるだけ簡潔にして頂き、使用環境、接続状況、エラーコードの内容などは具体的にお書き下さい。

当方では、最新のFAQを下記サイトで公開しております。よくアルご質問はFAQにまとめておりますので、予めこちらもご確認頂けますようお願い致します。

http://www.microtechnica.tv/faq/faq.cgi

マイクロテクニカ
microtechnica
〒158-0094　東京都世田谷区玉川1-3-10
TEL: 03-3700-3535　FAX: 03-3700-3548